UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Aplicación de control de cuentas bancarias multi entidad

Autor: Javier Martín González
Tutor: Eduardo Cermeño Mediavilla
Ponente: Juan Alberto Sigüenza Pizarro

mayo 2019

Aplicación de control de cuentas bancarias multi entidad

Javier Martín González

IMPRESO EN ESPAÑA

A mi familia, gracias

Dos clases de personas fracasan en la vida:
Aquellas que no saben nada y aquellas que creen saberlo todo

Warren Buffett

Agradecimientos

Quisiera agradecer este TFG especialmente a mis padres por el esfuerzo que habéis hecho a lo largo de toda mi vida para que hoy día haya podido llegar a escribir esta memoria, y a toda mi familia que habéis sido un apoyo para mi en los momentos buenos y en los malos.

A mi tutor Eduardo Cermeño por lograr que salga adelante este trabajo y lo aprendido contigo al haberme enseñado también tu visión en otros aspectos profesionales.

A Eduardo Hilario que durante todo el grado no solo has sido mi fiel compañero de prácticas y del que he aprendido un montón, sino también un gran amigo con el que he compartido momentos fantásticos.

A Inés Fernández, que desde el primer día que entré en la EPS has sido una buena amiga y una compañera clave para sacar adelante alguna que otra asignatura que se nos atascaba.

No puedo olvidarme de la gran familia que hemos formado en la DEISI, ahora renombrada a CODE, el buen rollo y las inolvidables tardes de actividades. Tampoco de todos los profesores que os habéis involucrado por mi aprendizaje haciendo que cada desafío se haya acabado convirtiendo en una pieza más del puzle de mi conocimiento.

Por último, fuera de la universidad os tengo que agradecer a todos los que en algún momento habéis formado parte de mi vida, sigáis o no conmigo y estéis cerca o en la otra parte del mundo. Sois muchas personas, nombres e historias que son imposibles de recoger en una página, pero sin vosotros no hubiera llegado a ser la clase de persona que soy ahora.

No obstante, entre todos quiero mencionar especialmente y siguiendo un orden cronológico por etapas de mi vida, a todos los profesionales de oncología del Hospital 12 de Octubre por lograr que hoy pueda seguir aquí, a Manuel Ajamil por haber confiado en mi para tus proyectos desde que yo tenía tan solo 13 años, a Jefferson Guaján por los momentos tan buenos que hemos pasado desde que nos conocimos en el colegio, a Manuel Jesús Martín porque aunque cuando te conocí todavía vivías en Jaén has sido siempre alguien muy cercano, a Marina Martín como vecina y compañera de entrenamientos que me hiciste amar el atletismo, a María Castro por ser mi alma gemela en nuestras tardes y fines de semana de salto de vallas, a Rodrigo Luján porque aunque aún sigas viviendo en tierras riojanas te podría considerar casi como mi hermano a la vez que compañero de profesión, a Irene Aguilera por ser esa amiga que apareciste en mi vida en el momento adecuado para lograr que venciera algunos de mis miedos y me hicieras ver las cosas de otra manera y a Enrique Sánchez como tutor de mis prácticas externas por todo lo aprendido de ti siendo claves tus consejos para este trabajo.

Gracias, mil gracias de corazón.

Resumen

El objetivo principal de este proyecto es la implementación de una aplicación de control de cuentas bancarias multi entidad, a la que se le ha denominado *GlobalBank*. Se trata de una aplicación web, con la que se pueden consultar los saldos y movimientos de los productos bancarios de diferentes entidades desde el mismo lugar.

Al inicio del proyecto se han hecho una serie de análisis sobre la nueva normativa PSD2 de regulación bancaria de la Unión Europea y cuáles son las maneras de poder obtener dicha información, para después centrarme en elegir las tecnologías apropiadas con la finalidad de que mi aplicación se adapte a todo tipo de dispositivos y que sea sencilla de utilizar por parte de cualquier persona, aún sin muchos conocimientos de navegación web.

Los frameworks utilizados para casi la totalidad del proyecto han sido Vue.js y Bootstrap. La base de datos utilizada en mi caso ha sido MySQL, pero también es plenamente compatible con PostgreSQL. Durante las fases finales del proyecto se han solicitado y recibido comentarios por parte de una serie de usuarios ajenos al desarrollo del proyecto y una vez implementada la funcionalidad completa, se han realizado una serie de pruebas para verificar la calidad y el buen funcionamiento del producto final.

Palabras clave

GlobalBank, aplicación web, Vue.js, Bootstrap, MySQL, JavaScript, PSD2, nueva regulación bancaria

Abstract

The main purpose of this project is to develop an application to manage bank accounts from different banking entities. The product is called *GlobalBank* and it's a web application that allows you to check balance and transaction history for accounts from different banks into the same place.

At the beginning of the project, I did an exhaustive research of PSD2, the new European Union payment service directive and the ways to get the users' bank information by myself. After that I looked for which technologies were availables to develop a responsive and adaptative design compatible with all devices and screens. I also made the effort to achieve my goal of creating an easy to use application for any user, even for those that don't have much web navigation experience.

The frameworks for almost all the project are Vue.js and Bootstrap. In my case I used a MySQL database, but the application is also compatible with PostgreSQL. In the last stages of the project I received some feedbacks from beta users alien to development. Finally, when the implementation was fully finished I accomplished some test to ensure the quality and good performance of the final product.

Keywords

GlobalBank, web application, Vue.js, Bootstrap, MySQL, JavaScript, PSD2, new payment services directive

Índice

1 Introducción **1**

1.1 Marco del proyecto 1

1.2 Motivación 2

1.3 Objetivos 2

1.4 Organización 3

2 Estado del arte **5**

2.1 Introducción 5

2.2 Servicios de agregación bancaria 5

2.3 Servicios de transferencia bancaria 6

2.4 Servicios de validación crediticia 6

3 Análisis **9**

3.1 Introducción 9

3.2 La normativa bancaria 9

3.2.1 PSD: Primera regulación bancaria de la Unión Europea 9

3.2.2 PSD2: La apertura completa de la industria bancaria en la Unión Europea 10

3.3 Posibilidades para acceder a la información bancaria 13

3.3.1 Bancos que si ofrecen estándares 13

3.3.2 Proveedores de acceso a la información bancaria 15

3.4 Roles de usuario 19

3.5 Casos de uso 20

3.6 Definición de requisitos 20

3.6.1 Requisitos funcionales 21

3.6.2 Requisitos no funcionales 21

3.7 Tecnologías para el desarrollo 22

3.7.1 Front-end 22

3.7.2 Back-end 24

3.7.3 El desarrollador web completo 26

3.8 Conclusiones del análisis 27

3.8.1 Tratamiento de la información 28

4 Diseño y desarrollo **29**

4.1 Introducción 29

4.2 Back-end 29
4.2.1 Conexiones con la API 29
4.2.2 Integración de la API 29
4.2.3 Mejoras de la API 31
4.3 Front-end 31
4.3.1 Bootstrap 31
4.3.2 Vue.js 31
4.4 Interfaz de usuario 34
5 Integración, pruebas y resultados **35**
5.1 Introducción 35
5.2 Funcionalidad 35
5.3 Usabilidad 36
5.4 Compabilidad universal 37
5.5 Servidores de pruebas 37
6 Conclusiones finales y a futuro **39**
6.1 Conclusiones finales 39
6.2 Trabajo futuro 40
Bibliografía **42**
Definiciones **43**
Acrónimos **45**
Apéndices **47**
A Guía del programador **49**
B Interfaz de usuario final **51**

Listas

Lista de figuras

3.1 Documentación de la API de Afterbanks . . . 15
3.2 Servicios PSD2 ofrecidos por Eurobits . . . 17
3.3 Presencia internacional de Salt Edge . . . 17
3.4 Diagrama de casos de uso de GlobalBank . . . 20
3.5 Distribución de los elementos por cuadrículas en Bootstrap . . . 23
3.6 Diferencias de rendimiento entre PHP 5.6 y PHP 7 . . . 26

4.1 Proceso de agregación de nuevo banco en Salt Edge Connect . . . 30
4.2 Ciclo de vida de un componente Vue.js . . . 32
4.3 Ejemplos de utilización del framework Vue.js . . . 33
4.4 Ejemplo de gráfico con Chart.js en base a la información de una cuenta bancaria . . . 34

A.1 Configuración de la base de datos y contraseña de la aplicación . . . 50
A.2 Configuración de los parámetros propios de Saltedge . . . 50
A.3 Configuración del cronjob en cPanel . . . 50

B.1 Visualización de interfaz global en pantallas grandes . . . 51
B.2 Visualización de interfaz de movimientos en pantallas grandes . . . 51
B.3 Visualización de interfaz de agregar banco en pantallas grandes . . . 52
B.4 Visualización de interfaz global en pantallas medianas . . . 52
B.5 Visualización de interfaz de movimientos en pantallas medianas . . . 53
B.6 Visualización de interfaz de agregar banco en pantallas medianas . . . 53
B.7 Visualización de interfaz global en pantallas pequeñas . . . 54
B.8 Visualización de interfaz de movimientos en pantallas pequeñas . . . 55
B.9 Visualización de interfaz de agregar banco en pantallas pequeñas . . . 56

Lista de tablas

3.1 Tabla comparativa de proveedores AISPs . . . 27

1 Introducción

1.1. Marco del proyecto

Antiguamente la gente tenía únicamente una cuenta corriente en un solo banco, mientras que ahora la gran mayoría tiene dos, tres o incluso más cuentas y en diferentes entidades. Nos encontramos en un entorno con una alta competencia bancaria en el que hay que destacar las recién llegadas *fintech* que mezclan las finanzas con la tecnología.

Estas empresas han crecido recientemente gracias a su gran apuesta por la innovación dentro del sector bancario a la vez que se centran en las necesidades del usuario [1]. Los aspectos a destacar son la inmediatez, la transparencia y el ahorro en comisiones, a cambio de convertir al cliente en un usuario digital que no necesita visitar las oficinas para sus rutinas.

Los bancos tradicionales han visto perjudicado su negocio y necesitan mucho tiempo para adaptarse a los nuevos tiempos de la forma que lo hacen las *fintech*, porque aunque su tecnología es muy segura, está instalada sobre sistemas antiguos que no puede sustituirse completamente de un día para otro [2].

Sin ir muy lejos, hasta el año 2007 las transferencias bancarias en España se realizaban mediante un sistema de compensación nacional, de forma que cuando se gestionaba un envío de dinero, de forma presencial o mediante un sistema electrónico, el dinero no era transferido al destinatario en ese momento. En su lugar, el banco de origen realizaba un apunte en nuestra cuenta bancaria reflejando dicha transacción, pero por motivos de coste solo se hacía el cierre de cuentas una vez al día. Al final de la jornada, se realizaba un ajuste de un banco con todos los demás bancos del país con el fin de realizar los correspondientes apuntes de ingreso en las cuentas de los clientes, y realizar el movimiento del dinero agrupando así miles de operaciones en solo una. Las transferencias nacionales podían tardar entre 1 y 4 días laborables en aparecer reflejadas en las cuentas de destino, y las internacionales hasta 7 días.

La firma por parte de España del Tratado de Adhesión a las Comunidades Europeas [3] y la adopción del euro [4] como moneda oficial, han logrado un marco de universalidad en toda la eurozona

tanto a nivel operacional como tecnológico.

No obstante, todo beneficio tiene sus inconvenientes y es de vital importancia lograr un correcto tratamiento de los datos, garantizar la privacidad de los mismos evitando que sean usados de forma inadecuada, y establecer si una determinada información es propiedad del banco o del usuario. Por todo ello, se han alcanzado una serie de consensos y regulaciones enfocadas tanto en la velocidad del procesamiento como en el derecho del acceso a la información.

1.2. Motivación

La razón que motiva la realización de este proyecto se debe al gran abanico de posibilidades que ha abierto la última regulación bancaria de la Unión Europea en el acceso a la información y en las ventajas para el usuario final.

Los datos en poder de los bancos son muy valiosos y con la nueva legislación se ha aclarado que los mismos son propiedad de los usuarios y no de los bancos, por lo que somos nosotros quienes tenemos la decisión sobre qué queremos hacer con ellos, a quién autorizamos el acceso a nuestra información bancaria y para qué fin se necesita.

Su puesta en marcha es interesante tratarla por ser un cambio que afecta directamente a las soluciones informáticas de terceros, ya que el banco debe permitir y facilitar ciertas acciones rompiendo de esta forma con la alegalidad presente hasta la fecha.

A raíz de ello, se amparan una gran serie de servicios como es el caso del acceso integrado a todas las cuentas bancarias de diferentes entidades desde un solo lugar por parte del usuario final, la facilidad para obtener consejos con respecto a las finanzas personales o la posibilidad de que empresas externas realicen sus estudios de viabilidad comercial y riesgo en base a una información objetiva.

1.3. Objetivos

El objetivo de este proyecto es desarrollar una aplicación web totalmente funcional en la que se puedan visualizar los saldos y las transacciones de las cuentas de una serie de bancos españoles. Se busca desarrollar algo sencillo para el usuario y accesible desde cualquier dispositivo, que garantice un fácil acceso por parte de todos los públicos a su situación financiera global, sean o no hábiles con la tecnología.

Para ello, se han planteado los siguientes hitos:

- Estudiar las regulaciones de acceso a la información bancaria y el consentimiento de los usuarios.
- Estudiar y analizar los proveedores de información bancaria que cumplan con la última normativa en materia de

acceso a los datos.

- Analizar las distintas tecnologías de desarrollo web (front-end y back-end) que permitan crear una aplicación para el usuario que sea sencilla y compatible con todos los dispositivos.
- Diseñar y desarrollar la aplicación según la regulación estudiada y con las tecnologías que más se adapten al producto que se pretende lograr.
- Realizar una serie de pruebas tanto a nivel de programación como de usabilidad con el fin de que el resultado se adapte a lo planteado.

1.4. Organización

La memoria consta de los siguientes capítulos:

- En el **capítulo 2**, se menciona con una serie de ejemplos una serie de aplicaciones que existen en el mercado con respecto al acceso y uso de datos bancarios para diferentes fines, y que sirven de base para este trabajo.
- En el **capítulo 3**, se describe principalmente el análisis de la normativa bancaria y las tecnologías existentes para el desarrollo del proyecto. También se toman decisiones en cuanto al almacenamiento y tratamiento de la información, la definición de requisitos y el diagrama de casos de uso.
- En el **capítulo 4**, se tratan los mecanismos realizados para el diseño y el desarrollo de la aplicación y cómo se han ido implementado las distintas tecnologías a lo largo del mismo.
- En el **capítulo 5**, se analizan las diferentes pruebas realizadas para garantizar el correcto funcionamiento de la aplicación y la respuesta ante diferentes casos que se pueden dar.
- En el **capítulo 6**, se detallan las conclusiones finales adoptadas a la finalización de este proyecto y los posibles cambios, mejoras y líneas de trabajo futuro.

2

Estado del arte

2.1. Introducción

Este proyecto de Trabajo de Fin de Grado (TFG) aborda el acceso a distintos bancos con el fin de crear una aplicación multi entidad de información financiera.

Para ello, en primer lugar, se detalla un estado del arte enfocado en conocer una serie de aplicaciones y servicios fintech para los que el acceso o la utilización de los datos bancarios de los usuarios son la fuente del negocio.

2.2. Servicios de agregación bancaria

Son servicios que permiten la consulta de los saldos y transacciones de productos financieros externos a la propia aplicación.

Bankia, BBVA, Openbank, Sabadell, Santander

Estos bancos ofrecen a sus clientes un servicio de agregación bancaria básico en el que poder consultar los datos y movimientos de otras entidades bancarias españolas de las que el usuario sea cliente.

El acceso a esta información está disponible dentro de la aplicación móvil de cada uno de estos bancos y no hay diferencias entre funcionalidades de uno u otro banco.

Fintonic

El portal Fintonic [5], nacido en el año 2011, tiene una amplia experiencia en el sector de la agregación bancaria y en un primer momento se caracterizó por el hecho de permitir consultar desde una misma aplicación el resumen financiero global al poder agregar diferentes bancos y entidades de crédito.

Poco a poco han ido incorporando más funcionalidades al servicio como un asesor financiero que realiza un análisis de los hábitos del usuario ofreciendo consejos, el estudio de los recibos cargados a la cuenta y su tipología con el fin de ofrecer alternativas que supongan un ahorro al consumidor o la posibilidad de recibir notificaciones por cobros de comisiones o transacciones duplicadas.

2.3. Servicios de transferencia bancaria

Son servicios que permiten realizar transferencias bancarias de una cuenta a otra, previa autorización del usuario, de una forma guiada y sin necesidad de introducir los datos bancarios de destino.

Klarna

Klarna [6] (anteriormente conocido como SOFORT) es una solución alemana que lleva funcionando desde el año 2005 en su país de origen y desde el 2011 en España. Es comunmente utilizado como una forma de pago alternativa a la tarjeta de crédito para hacer compras en páginas web que ofrezcan esta opción.

El éxito de este sistema se encuentra en que no es necesario registrarse en ningún sitio ni en disponer de algo especial, ya que tan solo son necesarias las credenciales para acceder a la banca online y la clave de firma de operaciones. Durante todo el proceso la plataforma guía al usuario por una interfaz que le va solicitando paso a paso una serie de datos (elegir el banco, introducir las credenciales de acceso, seleccionar cuenta de origen y confirmar la operación con la clave de firma).

Tras ello, y sin la necesidad de conocer ni indicar ningún dato del destinatario, la transferencia es ejecutada y el cargo es hecho de manera instantánea a la cuenta bancaria escogida sin ninguna comisión extra diferente a la que pueda cobrar el banco.

Trustly

Trustly [7] es la alternativa sueca nacida en el año 2008 que funciona de la misma manera que Klarna, diferenciándose de la anteriores por el hecho ventajoso de ofrecer más opciones de pago además de con las cuentas bancarias tradicionales, como es el caso de las cuentas de dinero electrónico PayPal o Skrill.

2.4. Servicios de validación crediticia

Son servicios que permiten tomar decisiones de riesgo y viabilidad comercial en base a los movimientos bancarios y capacidad de ahorro del usuario.

Instantor

Instantor [8] es un novedoso sistema de scoring de origen sueco que permite realizar de manera online un análisis de solvencia de un cliente obteniendo la respuesta en cuestión de segundos. La ventaja de esta plataforma se encuentra en la eliminación por completo de todo tipo de comprobación manual de nóminas y libretas.

Su funcionamiento es muy sencillo, y tan solo requiere que el usuario introduzca las credenciales de acceso de todas las cuentas bancarias que éste tenga para que el sistema proceda a recopilar la información y hacer el estudio de viabilidad.

3 Análisis

La aplicación a desarrollar la he bautizado como GlobalBank.

3.1. Introducción

En este apartado del trabajo, se va a realizar un análisis del proyecto, la normativa bancaria de acceso a la información y las tecnologías disponibles. Dado que la idea de la aplicación es que pueda ser utilizada por cualquier persona, es muy importante realizar una buena captura de los requisitos que deben de cumplirse antes de empezar a su programación, ya que un error en esta etapa conlleva que luego sea más costoso remediarlo.

3.2. La normativa bancaria

3.2.1. PSD: Primera regulación bancaria de la Unión Europea

El primer paso desde la Unión Europea con el fin de agilizar los pagos entre los países miembros de la eurozona, vino de la mano de Payment Services Directive (PSD) , la primera regulación bancaria entre países [9]. El principal objetivo era el de lograr una universalidad y estandarización.

Así nació Single Euro Payment Area (SEPA) , la estandarización a nivel europeo, con las siguientes características [10]:

- Objetivo: Creación de una zona en la que los pagos en euros se realicen en igualdad de condiciones, derechos y obligaciones.
- Participantes: 33 países, los cuales abarcan los 28 países de la Unión Europea, además de Islandia, Liechtenstein, Mónaco, Noruega y Suiza.
- Una transferencia a otro país miembro tiene que ser igual de fácil, rápida y costosa que una nacional dentro del mismo país.
- Todas las cuentas bancarias se identifican por su Internacional Bank Account Number (IBAN) , por lo que, si se vive o trabaja fuera de España, no es necesario más que una sola numeración para recibir ingresos o domiciliar

recibos, siendo válida en todo el territorio de la zona.

Este fue solo el primer paso de lo que vendría después. En el caso de España, se empezaron a adaptar los sistemas desde el año 2009, siendo la fecha máxima de implantación el 1 de febrero de 2014.

3.2.2. PSD2: La apertura completa de la industria bancaria en la Unión Europea

Mientras que todos los países miembros se iban adaptando a la regulación PSD , la Comisión Europea en el año 2013 ya empezó a proponer una revisión de la misma, la Revised Payment Services Directive (PSD2) , que pretendía profundizar en los elementos ya reglados de la normativa anterior para adaptarlo al sector digital y que los consumidores y negocios puedan beneficiarse de una mayor sencillez en el acceso a la información y en los pagos dentro del comercio electrónico [11].

La base principal de la PSD2 es la apertura de los servicios de pago a terceras empresas, poniendo de manifiesto quién es el dueño de los datos y cómo se deben ofrecer estos servicios siempre bajo la máxima garantía de confidencialidad en el tratamiento de la información y seguridad [12]. Anteriormente a esta normativa ya había empresas que accedían a toda la información del usuario como se analizará más adelante, sin embargo, todo ello se hacía dentro de un marco alegal sobre el que varios bancos ponían puertas.

Dentro de esta segunda regulación se obtuvo la posición de que toda la información personal y financiera de los clientes bancarios pertenecen al usuario y no a las entidades bancarias. De esta forma, el usuario es el único que tiene el control de decisión en cuanto al hecho a quién quiere facilitar esta información y cómo quieren gestionar los permisos con las diferentes empresas que requieran el acceso [13]. Este escenario es algo complejo ya que es una normativa a nivel de la Unión Europea que debe permitir que una misma empresa debe de poder tener acceso a la información de clientes españoles, franceses, holandeses, etc., independientemente de donde se encuentre establecida su matriz.

Aunque dentro de la normativa no se ha establecido de forma explícita cómo debe hacerse el acceso a la información, la mayoría de los profesionales del sector tecnológico y financiero dieron por hecho que se realizaría mediante una Application Programming Interface (API) debido su sencillez y estandarización en la interconexión de sistemas; dejando atrás a métodos más rudimentarios como el *screen scraping* como una técnica para extraer información de sitios webs simulando la navegación de un humano, y que incluso en un primer momento llegó a considerarse ilegal.

Como en la época en la que vivimos los datos son muy valiosos y no se ha acordado en ningún momento un mínimo en cuanto a nivel de definición, nomenclatura, protocolo de acceso o autenticación,

se especuló sobre la posible picaresca de los bancos en cuanto a ofrecer una API de baja calidad o con una mala garantía Service Level Agreement (SLA) que pueda impedir obtener la información deseada. Para estos casos, tras un tira y afloja entre las *fintech* y los bancos tradicionales, la Comisión Europea reculó y optó por considerar el *screen scraping* como una técnica totalmente legal cuando la vía oficial no responda como debería o si se detecta una mala fe por parte del banco [14] [15].

Igualmente, se está creando un marco en cuanto a la velocidad de las transferencias de forma que se dejaría atrás el actual sistema de compensación bancaria para empezar a sacar provecho a las nuevas tecnologías y sistemas que permiten que las transferencias entre dos cuentas bancarias con IBAN europeo de hasta un importe de 15.000€, queden realizadas, confirmadas y reflejadas en un máximo de 10 segundos desde la ejecución. Estas transferencias inmediatas funcionarán las 24 horas del día, cualquier día del año, incluso en festivos. Este punto es clave para lograr una mejora en el comercio electrónico además de repercutir positivamente en los costes. No obstante, en el aspecto costes el cambio se espera a medio plazo, ya que como todavía no está implementado por parte de todas las entidades, lo consideran una característica premium de la que sacar partido, pero el coste real para el banco es de apenas unos céntimos por operación, mucho inferior al de las transferencias tradicionales.

Dentro de PSD2 , la Autoridad Bancaria Europea (ABE) ha establecido una serie de figuras [16] que se analizan a continuación y que son importantes para comprender el planteamiento final de mi aplicación.

AIS: Servicios de información de cuenta

Dentro de esta categoría se engloban los distintos servicios que recogen y almacenan la información de las distintas cuentas bancarias de un cliente desde un solo lugar, de tal forma que se les permite tener una visión global de su estado financiero desde donde poder realizar un análisis y recomendaciones según sus gastos, ingresos, patrones o necesidades financieras.

PIS: Servicios de iniciación de pagos

La introducción de los Payment Initiation Service (PIS) es una de las grandes novedades de la puesta en marcha de PSD2 y es que no solo se habla de poder facilitar la información del estado financiero a terceras empresas como en el caso anterior, sino que también debe ser posible que estar terceras empresas, previa autorización del cliente, puedan realizar una transferencia desde la cuenta de un cliente hacia la cuenta de un vendedor a través de la API sin necesidad de que intervengan terceras entidades de procesamiento de pagos.

El funcionamiento actual a la hora de hacer un pago electrónico es el siguiente:

1.– El usuario quiere hacer un pago en un comercio

2.– El comercio recibe la petición y se coordina con su proveedor de pagos

3.– El proveedor de pagos se encarga de tratar con el usuario

4.– El usuario realiza el pago con la tarjeta de crédito proporcionada por su banco en nombre del emisor de la tarjeta (Visa, Mastercard, etc)

5.– Se procesa el cobro a la tarjeta a través de la entidad emisora de la tarjeta

6.– El dinero llega al banco del comercio

Todas las comisiones indicadas, son pagadas por el comercio. Con los nuevos mecanismos a través de una API que haga de un servicio PIS , se eliminaría el punto 3 por completo, la comisión del punto 4 (ya que se espera que los usuarios tengan las transferencias SEPA totalmente gratuitas), el punto 5 por completo y la comisión que le es cobrada al comercio en el punto 6 por parte del emisor de la tarjeta. Por lo tanto, la mejora se produce tanto en tiempo al ser realizado el pago de forma más rápida y con recepción instantánea, y los costes asociados para el comercio, pudiendo ser así más competitivos.

AISPs: Proveedores de servicios de información de cuenta

Para poder ofrecer un servicio Account Information Service (AIS) no es necesaria ninguna autorización por parte de organizaciones bancarias, sin embargo, para poder acceder directamente a dicha información bancaria, con el fin de garantizar la máxima seguridad del proceso se ha establecido la figura del proveedor de servicios de información de cuenta, el cuál es una figura externa a las entidades bancarias.

Estos proveedores tras realizar la correspondiente solicitud ante el organismo bancario oficial y una vez corroborado que se cumplen todos los requisitos de PSD2 , serán los encargados de obtener una licencia para acceder mediante la API oficial de cada banco, a la información bancaria de los clientes que así lo autoricen previamente. Su fin puede ser para utilizar ellos mismos la información o con la posibilidad de cedérsela a terceras personas como parte de su negocio, evitando de esta forma que cualquier entidad o persona, sin reunir unos requisitos mínimos ni pasar por auditorías de seguridad y calidad, obtenga datos que puedan ser comprometidos y utilizados con intenciones no acordes a la normativa.

PISPs: Proveedores de servicios de iniciación de pagos

En la otra cara de la moneda, los Payment Initiation Service Providers (PISPs) los proveedores externos que se encargan de ofrecer los servicios de iniciación de pagos, los cuales no solo podrán actuar como Account Information Service Providers (AISPs) con acceso de solo lectura, sino que son capaces también de iniciar una operación de pago en nombre de un tercero hacia una cuenta bancaria de destino fijada.

Para poder realizar dicha transacción, deben contar con la correspondiente autorización del cliente

que le es otorgada en el mismo instante de la petición, debiendo ser un doble factor adicional a la contraseña de acceso a la banca online (como puede ser una posición de una tarjeta de coordenadas, clave de firma, un código SMS de un solo uso, etc.).

De igual manera, estos proveedores pueden ofrecer sus servicios a terceros que no hayan solicitado directamente una autorización al organismo oficial.

3.3. Posibilidades para acceder a la información bancaria

A la hora de la realización del TFG , habiendo superado la fecha máxima de implantación de PSD2 impuesta por la Unión Europea para septiembre de 2018, la realidad es la gran mayoría de los bancos no han llegado a tiempo a ofrecer su API oficial de conexión de la forma que estaba prevista.

Incluso entre los pocos bancos que si tienen públicas sus documentaciones y definiciones, nos encontramos con un unas posibilidades muy básicas y basadas en una *sandbox* con datos ficticios para ir abriendo el camino al futuro. Por la parte de los proveedores de acceso a la información, si que es posible encontrar más preparación y cumplimiento de las medidas de PSD2 .

3.3.1. Bancos que si ofrecen estándares

En otros países como el Reino Unido, desde hace bastantes meses si se han llegado a formar agrupaciones de bancos para ofrecer una API universal que si que llegó a tiempo a los plazos marcados por PSD2 , sin embargo en España el alcance ha sido muy reducido.

En un primer momento cada banco iba a ofrecer sus conexiones de forma individual y que hubiese que recurrir a cada uno de ellos para obtener el acceso, para lo cual se empezaron a preparar dos grandes bancos españoles que se analizan en este apartado. Sin embargo, y aunque a la fecha de escribir esta memoria todavía no hay nada funcional, se está desarrollando en España un proyecto de API universal que va a conectar a todos los bancos españoles.

La empresa responsable de llevar a cabo esta agrupación es Redsys, la cual cuenta con una gran cantidad de experiencia en el sector bancario de nuestro país, siendo la pasarela por excelencia para el pago con tarjeta en comercio electrónico al hacer de intermediarios entre Visa y Mastercard con cada banco y tienda online; y además su red de transacciones es usada por los datáfonos de casi la totalidad de las tiendas físicas de nuestro país a la hora de procesar los pagos, devoluciones o recargas de móviles, entre otros servicios.

BBVA

Un banco pionera cuanto a avances tecnológicos es el BBVA, y ya no solo en tecnologías novedosas y pioneras dentro de su servicio bancario (como es el caso de la video identificación remota, la renovación de cajeros, las posibilidades de la aplicación) sino que desde que el año 2016 abrieron de forma pública el llamado BBVA API Market [17], que aunque en fase piloto, su intención era la de apostar por una plataforma abierta con la que terceras empresas pudieran desarrollar sus negocios en base a los servicios del banco.

Dentro del BBVA API Market, es posible encontrar documentación API tanto para ofrecer un servicio de información de cuentas como un servicio de iniciación de pagos. He podido registrar mi aplicación sin ningún inconveniente, pero lo único a lo que he podido tener acceso es a un *sandbox* con unas credenciales de prueba ya que para acceder a un servicio real es necesario atravesar un proceso de revisión de la aplicación en cuestión.

Curiosamente, desde BBVA tienen tan abierta la mente que ellos mismos en los artículos de su blog [18] informan sobre la PSD2 y las aplicaciones fintech, y además te indican una serie de proveedores AISPs que conectan perfectamente con el banco y con los que incluso ellos mismos han colaborado en algún caso para lograr la integración. Parte de estos son analizados en el siguiente apartado siendo algo que ha resultado vital de cara a tomar una decisión sobre la implementación de mi aplicación y en consecuencia la viabilidad de este TFG .

ING Direct

Para este banco de origen holandés 100 % operativo en España, también es posible encontrar un portal de desarrolladores, el ING Developer Portal [19] en el que ofrecen un *sandbox* para un servicio de información de cuentas como para un servicio de iniciación de pagos.

Sin embargo, accediendo a cualquiera de estas dos opciones nos indica que la última versión de la API ofrecida está en estado “Deprecated”, y aunque también me ha sido posible obtener un token de aplicación tras registrar la mía en su *sandbox*, los certificados de seguridad que se ofrecen para poder probar esta *sandbox* están caducados.

Dentro del portal de desarrolladores de ING, es posible encontrar una Frequently Asked Questions (FAQ) con información de cómo el banco se está preparando de cara a PSD2 , y en ella indican que las especificaciones finales serán ofrecidas por ING en el tercer trimestre de 2018, y que el servicio completo sin *sandbox* funcionará en el cuarto trimestre de 2018. Sin embargo, mientras escribo estas líneas nos encontramos en el segundo trimestre de 2019 y la página sigue mostrando dicha información anticuada, que no ha sufrido ningún cambio ni actualización desde al menos el tercer trimestre de 2018 cuando yo accedí por primera vez.

3.3.2. Proveedores de acceso a la información bancaria

Dentro de las investigaciones que he realizado de cara a mi TFG , he encontrado una serie de proveedores, que actuando como AISPs , ofrecen a aplicaciones de terceros a través de una API propia la información real sobre las cuentas bancarias indicando el usuario y contraseña de acceso de lectura a la banca electrónica. Estos proveedores son capaces de facilitar los datos de saldos y transacciones, y continuación se enumeran y analizan.

Afterbanks

Afterbanks [20] fue el primer proveedor que encontré cuando empecé a investigar sobre las posibilidades para acceder a los datos bancarios, siendo además una empresa española con la que la comunicación es fácil con el fin de descubrir cómo funciona.

Al acceder a su página web descubrí que tienen dos productos: ellos mismos tienen una aplicación similar a la que yo desarrollo en este proyecto, aunque con un diseño no muy adaptado a los tiempos actuales, y ofrecen una API marca blanca para que terceros puedan desarrollar un sistema similar. La documentación de su API es muy sencilla y con apenas un par de verbos se puede acceder al listado de bancos compatibles y a la información de saldos y transacciones (figura 3.1).

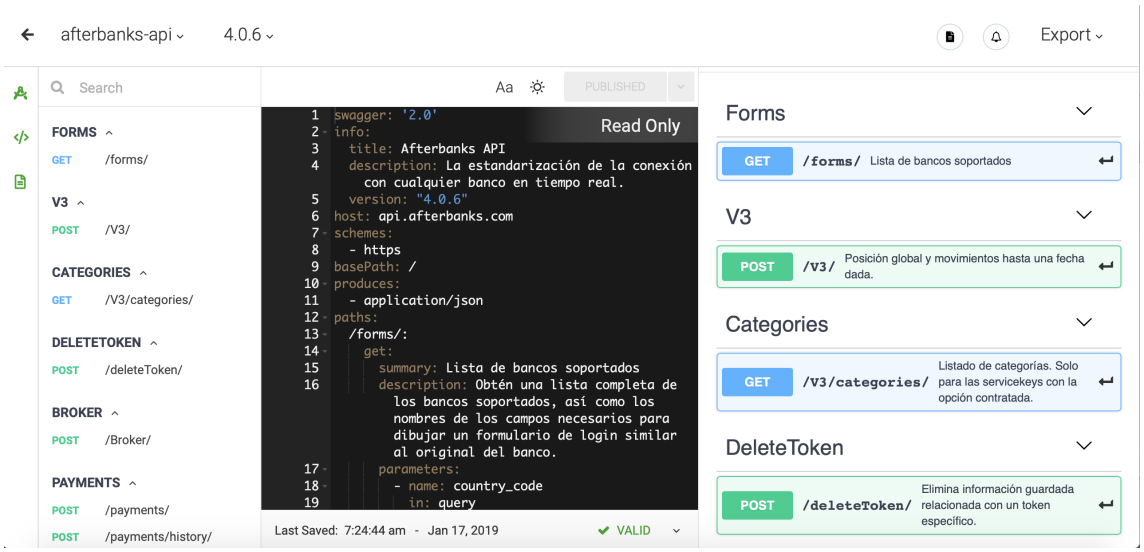

Figura 3.1: Documentación de la API de Afterbanks

Esta empresa tiene como ventaja que la implementación es muy sencilla en un par de llamadas y con un trabajo por detrás bastante profesional de cara de lograr una universalidad entre bancos a la hora de devolver los datos JSON y además al ser una empresa local se centra principalmente en bancos españoles que es el enfoque que en mi caso estoy interesando en abarcar y son compatibles con el 100 % de los bancos nacionales y algún que otro de Sudamérica.

Sin embargo al ponerme en contacto con ellos, no tienen ninguna sandbox con datos falsos para

pruebas con los que poder conectar mi aplicación. Cuando solicité el acceso al entorno real, me mandaron un folleto con más información sobre las posibilidades de trabajar con ellos junto a casos de éxito de varias empresas que obtienen los datos de ellos, acompañado de un presupuesto de varios cientos de euros al mes con un límite de 1500 llamadas a su API por mes natural.

Eurobits

Eurobits [21] es una empresa española, fundada entre varios bancos y especializada en todo tipo de productos financieros de distinta índole, que además ofrece servicios PSD2 para el acceso a la información bancaria 3.2.

Conocí de su existencia gracias a Youtube al visualizar la conferencia ”PSD2: Agregar o ser agregado” [22] que impartieron en noviembre de 2018. Es una charla muy interesante en la que aprendí una gran cantidad de cuestiones a nivel técnico y legal que han sido claves para las decisiones de este TFG .

Al visualizar su página web se puede ver que tiene una gran cantidades de casos de éxito de implementaciones de productos en grandes entidades bancarias (Santander, Sabadell, Openbank y Bankia), del agregador bancario Fintonic que utiliza su API para obtener todos los datos que luego analiza e incluso con el Gobierno de la Rioja. Entre sus colaboradores también se encuentran grandes tecnológicas como Telefónica y Accenture.

Dentro de un artículo especial para PSD2 en su blog [23], anuncian que el propio servicio de agregación bancaria del BBVA, que se ha anunciado en televisión a bombo y platillo con el lema ”Tus otros bancos son bienvenidos” [24], funciona por detrás por los datos que Eurobits ofrece. Por lo tanto, se deduce que es un gran proveedor con unos clientes a los que tiene que proporcionar una muy buena calidad.

En cuanto a productos PSD2 tienen una categoría especial para ello, en la que se ofrecen tanto como AISPs ofreciendo a terceros el acceso a los datos y como desarrolladores en la que ellos mismos son los encargados de definir y configurar una API bancaria de calidad. En su FAQ se venden como un proveedor con una API muy universal y un JSON muy fácil de interpretar, que además permite obtener una categorización de cada movimiento.

Sin embargo, me intenté poner en contacto con ellos en varias ocasiones por e-mail y redes sociales solicitando más información o la existencia de una sandbox y nunca obtuve respuesta.

Salt Edge

Salt Edge [25] es un proveedor AISPs de origen alemán con un gran recorrido internacional presente en bastantes países ofreciendo una información bancaria muy completa (figura 3.3).

Figura 3.2: Servicios PSD2 ofrecidos por Eurobits

Figura 3.3: Presencia internacional de Salt Edge

Aunque en España solo tiene 27 accesos bancarios (frente a los más de 3000 de Alemania) y un par de ellos se podrían considerar repetidos (ya que separa entre acceso con credenciales o lectura mediante fichero Comma Separated Value (CSV) que permiten exportar determinados bancos), todos los grandes bancos nacionales son compatibles e incluso cuentan con alguna que otra entidad más pequeña.

Los datos de las credenciales de los usuarios para una aplicación que no ha pasado un proceso de auditoría, deben ser recogidos directamente con una llamada a la plataforma de Salt Edge, que desde su propio dominio y servidor, solicitan al usuario los datos, los procesan y devuelven a la aplicación en cuestión un token con el que poder acceder luego. Para aplicaciones en un entorno real que han pasado una serie de revisiones, también se da la opción de enviar directamente las credenciales por la API .

Hay que destacar que tiene una API propia muy bien documentada, que aunque no es la más fácil de implementar e incluso algunas funciones me envían comandos de retorno como si yo les ofreciera a ellos una API de conexión hacia mi aplicación, está bastante dividida por módulos acordes a todas las opciones posibles. La pasarela externa de introducción de datos está disponible en varios idiomas, entre ellos el español, y además hay una pequeña comunidad de usuarios en la que se resuelven problemas o se ayuda a entender algún aspecto de la integración.

A nivel comercial, ofrece un acceso gratuito con unos bancos simulados de prueba sin tener que registrar la aplicación en su sistema. Para un entorno de acceso a bancos reales, también se ofrece una opción gratuita tras registrar la aplicación, con un número limitado de credenciales almacenadas (aplicaciones de uso personal en la que cada uno aloja su aplicación y tiene su propia licencia de conexión para sus propias cuentas) y un plan de pago para un número ilimitado de credenciales en la que se pretende gestionar las cuentas de varios usuarios distintos y en las que una empresa es quien ofrece el producto en sus servidores sin dar la aplicación completa como tal al usuario que la va a utilizar.

Plaid

Plaid [26] es otro de los proveedores dentro de la categoría de los AISPs y con una gran experiencia en el sector.

Conocí la existencia de éste gracias al artículo escrito por BBVA informando acerca de la PSD2 y sus aportaciones desde el principio de la ley. Incluso desde la propia página del banco te explican cómo es el proceso de configuración con este operador, ya que al parecer colaboraron mucho para lograr una integración de calidad y en tiempo real, basado en una API sencilla hecha por desarrolladores para desarrolladores.

La conexión con las credenciales del usuario también se hace a través de su propia plataforma,

creando una serie de claves públicas y privadas con las que mi aplicación podría obtener la información de saldos y transacciones, sin manejar en ningún caso contraseñas que pudieran ser comprometidas.

Aunque también cuenta con una versión gratuita para uso personal, una desventaja bastante grande de este proveedor es que las únicas entidades que permite y que trabajan en España son BBVA y American Express. Todavía necesita expansión en nuestro país.

Tink

Tink [27] es un proveedor que ha entrado recientemente en España a ofrecer los servicios de agregación y aunque pisando fuerte y expandiéndose rápidamente, todavía se encuentra en una fase muy temprana.

Su nicho de clientes ha sido Reino Unido desde 2012 aunque con la implantación de PSD2 tienen muchos planes a futuro en diversos países europeos. A la hora de empezar la investigación para este TFG no operaban en España, sin embargo en febrero de 2019 recibí un email en el que me invitaban a una fase de pruebas en la que se me daba acceso para probar diferentes bancos españoles y además se ofrecía una pequeña gratificación a cambio de responder después a unas preguntas sobre el funcionamiento o fallos encontrados.

Su idea es la de ofrecer una API sencilla, completa y muy bien documentada para que cualquier desarrollador sea capaz de implementarla sin tener mucha experiencia en el sector y sin tener que hacer malabares con los verbos para entender la modularidad. La conexión con las claves bancarias, al igual que en Salt Edge y Plaid, se hace mediante su plataforma externa que genera un token para luego poder acceder yo sin tratar dichos datos.

Las desventajas de ser una fase temprana es que la interfaz externa de introducción de datos todavía no ha sido traducida al español y que los bancos españoles compatibles con muy limitados, habiendo incluso grandes entidades que no son conectables aún.

A nivel comercial, ofrece un acceso gratuito con un número limitado de entidades agregadas (para aplicaciones de uso personal en la que cada uno aloja su aplicación y tiene su propia licencia de conexión para sus propias cuentas) y planes de empresa que van desde los 500€ al mes.

3.4. Roles de usuario

Este proyecto está pensado para el uso por parte de un usuario final, por lo tanto éste es el único ente físico que interactúa con el sistema. El usuario es el encargado de agregar o eliminar los bancos y cambiar una serie de ajustes a su antojo.

En cuanto a otros participantes, se encuentra el proveedor externo (el AISPs) que es quien se

encargará de la parte de gestión de permisos a las entidades bancarias y de las actualizaciones de la información que devuelve la API , pero es un agente ajeno que no interviene para nada en ningún aspecto de esta aplicación en concreto.

3.5. Casos de uso

La idea del desarrollo de la aplicación de control de bancos multi entidad es el de crear algo sencillo y visual para el usuario, sin sobrecargas por elementos o funcionalidades más allá de las necesarias para el fin para el que ha sido creado. El diagrama de casos de uso puede visualizarse en la figura 3.4.

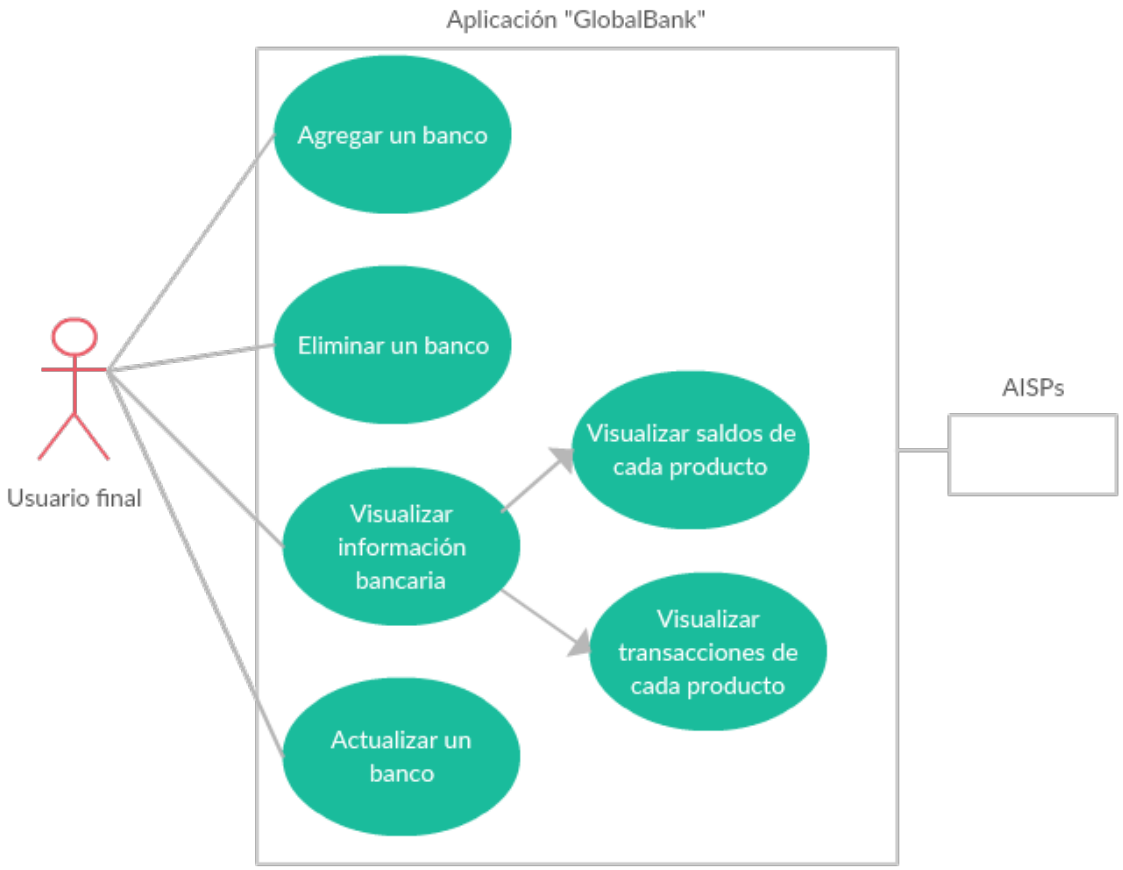

Figura 3.4: Diagrama de casos de uso de GlobalBank

3.6. Definición de requisitos

Los requisitos se subdividen en dos categorías:

- Requisitos funcionales
- Requisitos no funcionales

3.6.1. Requisitos funcionales

Los requisitos funcionales que deben cumplirse se enumeran a continuación:

RF-1.– El control a la aplicación debe estar protegido por una contraseña

RF-2.– Debe poder cambiarse la contraseña de acceso al sistema

RF-3.– Debe ser posible agregar un banco al sistema

- **RF-3.1.–** Debe ser posible agregar las principales bancarias españolas
- **RF-3.2.–** Debe ser posible agregar las principales entidades de crédito españolas

RF-4.– Debe ser posible eliminar un banco del sistema

RF-5.– Debe ser posible actualizar la información de un banco agregado al sistema

- **RF-5.1.–** Debe ser posible optar por una actualización de información manual
- **RF-5.2.–** Debe ser posible optar por una actualización de información automática una vez al día

RF-6.– Debe ser posible conocer el saldo de cada producto de cualquier banco agregado al sistema

RF-7.– Debe ser posible distinguir mediante a simple vista mediante iconos y colores el tipo de producto y el saldo (a favor o en contra) de dicho producto de cualquier banco agregado al sistema

RF-8.– Debe ser posible conocer el saldo global teniendo en cuenta la totalidad de los productos de un mismo banco agregado al sistema

RF-9.– Debe ser posible conocer el estado financiero general teniendo en cuenta el saldo global de la totalidad de los bancos agregados al sistema

RF-10.– Debe ser posible visualizar las transacciones de cada producto de cualquier banco agregado al sistema, siempre y cuando haya movimientos disponibles

RF-11.– Debe mostrarse dentro de la página de transacciones de cada producto un gráfico informativo de los movimientos (ingresos o gastos) de cada producto para los últimos 30 días

3.6.2. Requisitos no funcionales

Los requisitos no funcionales esperados se enumeran a continuación:

RNF-1.– Debe ser accesible desde un navegador web, garantizando como mínimo la correcta funcionalidad accediendo desde Google Chrome, Mozilla Firefox, Microsoft Edge y Safari en sus últimas versiones

RNF-2.– Debe ser una web responsive, que se adapte a todas las pantallas y tipos de dispositivo de forma nativa

RNF-3.– El consumo de recursos debe ser moderado, para garantizar que sea accesible incluso desde terminales de gama más baja

RNF-4.– Para completar cualquier acción, debe poder hacerse en un máximo de 3 clics

RNF-5.– Se debe garantizar la correcta protección de la información transferida, para lo que se exige contar con un certificado de seguridad Hypertext Transfer Protocol Secure (HTTPS) que garantice la confidencialidad frente a ataques de captura de paquetes

RNF-6.– Debe ser compatible con varios tipos de bases de datos, como mínimo debe tener compatibilidad con MySQL y PostgreSQL

3.7. Tecnologías para el desarrollo

Para la realización de la aplicación de acceso a saldos bancarios multi entidad, he optado por un desarrollo web. Hoy en día, cualquier tarea que queramos realizar en cierta parte tiene implicada una parte de web, y es que, aunque la primera página web apareció en el año 1990 de la mano de Tim Berners-Lee, los avances que se han producido desde entonces han sido grandes.

Esta primera web, se puso en funcionamiento dentro del dominio del European Organization for Nuclear Research (CERN) y se considera como el nacimiento del World Wide Web, caracterizándose por su fondo blanco, texto negro e hipervínculos de color azul y subrayados. Fue una versión muy básica de lo que hoy en día somos capaces de hacer.

Tras ello, nació la Web 2.0 o más conocida como la web social, la cual, aunque fue bautizada así en el año 1999, no fue hasta el año 2005 cuando Tim O'Reilly definió qué era realmente. Esta versión de la web tuvo un gran boom gracias al crecimiento de las redes sociales, pero sin embargo a día de hoy también se la puede considerar como algo del pasado.

Hoy en día, nos movemos en un universo de la Web 3.0 que busca una compatibilidad universal sin depender de ningún dispositivo en concreto para acceder a toda su funcionalidad. Las principales características de este modelo se basan en un diseño intuitivo, de fácil uso, minimalista y adaptada a todo tipo de personas, independientemente de sus conocimientos informáticos.

Los dos motivos por los que finalmente me he decantado por un servicio web han sido en primer lugar, debido a su simplicidad de acceso desde cualquier ubicación (ya sea a través de un dominio web o por localhost con su correspondiente base de datos local) y en segundo lugar por su posibilidad de adaptación desde un mismo punto a todo tipo de dispositivo (ordenador, móvil, tableta, televisión, etc) sin perder funcionalidades individuales que pudieran ser propias de cada sistema en caso de diseñarse a través de una aplicación nativa.

3.7.1. Front-end

Dentro de una aplicación, cuando hablamos de front-end nos estamos refiriendo a la parte visual, que en resumen es la forma en la que se le muestran los datos al usuario y cómo éste interactúa con ellos.

Una aplicación se considera responsive cuando es capaz de adaptarse a cualquier dispositivo, ofreciendo una visualización e interfaz acorde al tamaño de éste sin necesidad de reducir el zoom de la página para interactuar con su contenido.

Por otro lado, se puede considerar un sistema reactivo a aquel en el que se actualiza la información que se debe visualizar en tiempo real en función de las acciones que realice el usuario, y sin necesidad

de tener que actualizar la página para ello. Con las nuevas tecnologías en diseño web es una tarea muy sencilla y limpia a nivel de código.

Bootstrap

Bootstrap [28] trata de un framework que fue liberado por Twitter en el año 2011 y que sirve para el diseño de páginas web de una forma rápida. La principal característica de Bootstrap es su cuadrícula basada en 12 columnas. Gracias a ello, cada elemento consta de 12 unidades siendo divisibles entre 2, 3, 4 y 6 como se puede ver en la figura 3.5). Esto logra mantener una vista muy limpia e uniforme, y que además se adapta fácilmente a cualquier dispositivo. Además de ello, Bootstrap contiene elementos tipográficos, formularios, botones, cuadros, menús de navegación y otros de diseño basados en HyperText Markup Language (HTML) y Cascading Style Sheets (CSS) .

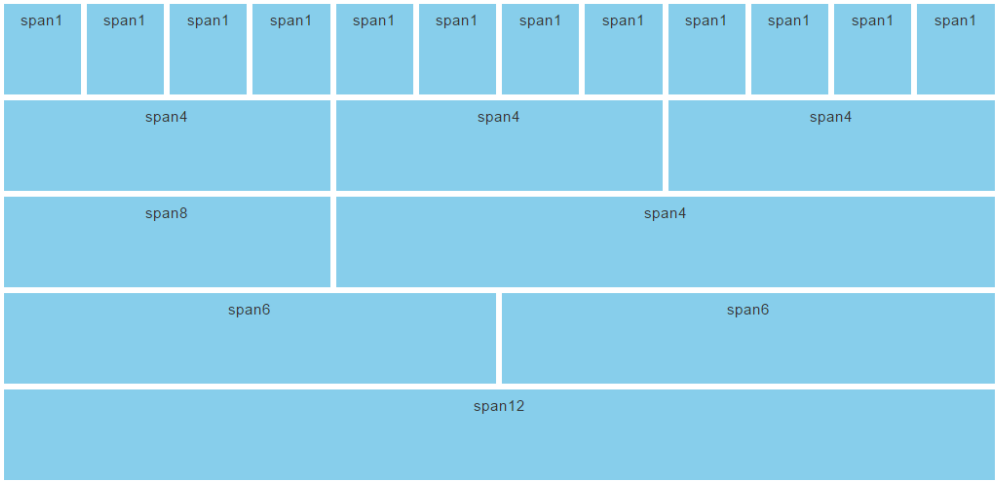

Figura 3.5: Distribución de los elementos por cuadrículas en Bootstrap

Otra de las ventajas es que es opensource y la curva de aprendizaje es muy rápida y que es posible encontrar una gran serie de temas para que la interfaz sea diferente a la estándar.

SASS y LESS

Syntactically Awesome Stylesheets (SASS) y Leaner Style Sheets (LESS) son dos precompiladores de CSS . Éstos permiten reescribir código de estilos de una forma más sencilla y fácil de leer, añadiendo una serie de características como es la herencia, operaciones y funciones. Esto no es posible hacerlo con un CSS nativo.

En el caso de Bootstrap v3 se utiliza SASS , sin embargo en Bootstrap v4 recientemente se ha optado por migrar su compilación a LESS

ReactJS

ReactJS [29] es un framework front-end que fue liberado por Facebook en el año 2013. Es de código abierto y se ha utilizado en grandes aplicaciones como la propia Facebook y su aplicación filial Instragram.

Se basa en compontentes con un ciclo de vida determinado con un funcionamiento de flujo de datos unidireccional, de forma que los componentes de orden superior son los que propagan la información a los de orden inferior. También hay que destacar que ReactJS no retorna código HTML , sino que son similar a funciones JavaScript con una sintaxis propia llamada JSX.

Vue.js

Vue.js [30] es otro de los frameworks para la parte visual de una aplicación web. Fue creado por Evan You, quien trabajaba en Google realizando prototipos. Es un lenguaje de muy reciente creación, en concreto del año 2014, sin embargo, su evolución y aceptación está siendo espectacular y la curva de aprendizaje es más rápida que la de otros mecanismos.

Entre las principales características de Vue.js tenemos las siguientes [31]:

- Es muy accesible.
- Es muy versátil, con un núcleo muy pequeño de tan solo 74 kilobytes, pero que se escala fácilmente a través de la adición de plugins.
- Es reactivo.
- Cuenta con una gran cantidad de usuarios colaborando con su núcleo a diaria y en constante crecimiento en las comunidades de desarrolladores.
- La licencia de uso concede al usuario final los derechos de usar, copiar, modificar, publicar, distribuir o sublicenciar el software

3.7.2. Back-end

Mientras que el front-end es la parte visible, la parte back-end es la cara oculta de la web que hace que todas las piezas y la lógica encaje. En esta parte de la programación, se siguen manteniendo las tecnologías más tradicionales.

En este escenario, es habitual la coexistencia de una parte front-end con tecnologías más modernas que se comunican en la parte back-end con otros modelos que llevábamos arrastrando desde años atrás.

Node.js

Node.js [32] es una librería de software de código libre, que basado en JavaScript, puede realizarse una combinación y ser válido tanto a nivel front-end como back-end, actuando a la vez tanto en cliente como en servidor.

Node.js se caracteriza por un buen rendimiento que con cada petición dispara la ejecución de evento para ser atendido directamente, frente a otros sistemas que crean un hilo por cada solicitud y pueden acabar bloqueando el servidor si hay mucha concurrencia o si se bloquean los procesos. La gran ventaja de esta librería es que para soluciones en la que la memoria es limitada, su rendimiento es bastante bueno.

Hay que destacar la existencia de numerosas extensiones libres que se pueden encontrar para ampliar la funcionalidad y que su motor ha sido desarrollado por Google.

PHP

Es normal que la gente te mire raro cuando les dices que estás programando en Hypertext Preprocessor (PHP) , no al nivel de otros lenguajes como si les dices que lo haces en Fortran o en Cobol pero parece que eres un viejuno.

Sin embargo, el Content Management System (CMS) más grande del mundo que es Wordpress está programado en PHP y está muy extendido por la red. Además, la gran mayoría de frameworks más populares están escritos en este lenguaje. Esto quiere decir que todavía se utiliza y mucho, e incluso plataformas como Tumblr han realizado recientemente su migración a este lenguaje.

Cuando PHP nació en el año 1995 de la mano de Rasmus Lerdorf, no estaba pensado como un nuevo lenguaje, sino como un simple script que utilizó el creador para contabilizar las visitas que recibía en su página web, basándose es ciertos parecidos con el lenguaje de programación C [33].

Desde esta primera versión, PHP ha estado en constante evolución y se ha reescrito en varias ocasiones aunque no fue hasta el año 2004 con la quinta versión del lenguaje cuando empezó a soportar la orientación a objetos. Si que es cierto que PHP en su versión 6 nunca llegó a ver la luz por sus grandes problemas, siendo un duro estancamiento del lenguaje donde más de uno proclamó su fin, pero se replanteó desde cero volviendo a resurgir con la séptima y actual versión.

En el momento que nos encontramos ahora, el rendimiento y el uso de la memoria ha mejorado notablemente (como se puede ver en la figura 3.6), y además se han resuelto importantes problemas de inconsistencia y de gestión de tipos.

El lenguaje de programación PHP tiene una curva de aprendizaje muy rápida y es libre, lo que ha conseguido lograr una gran comunidad de usuarios y una configuración muy sencilla que se adapta a todo tipo de desarrollos.

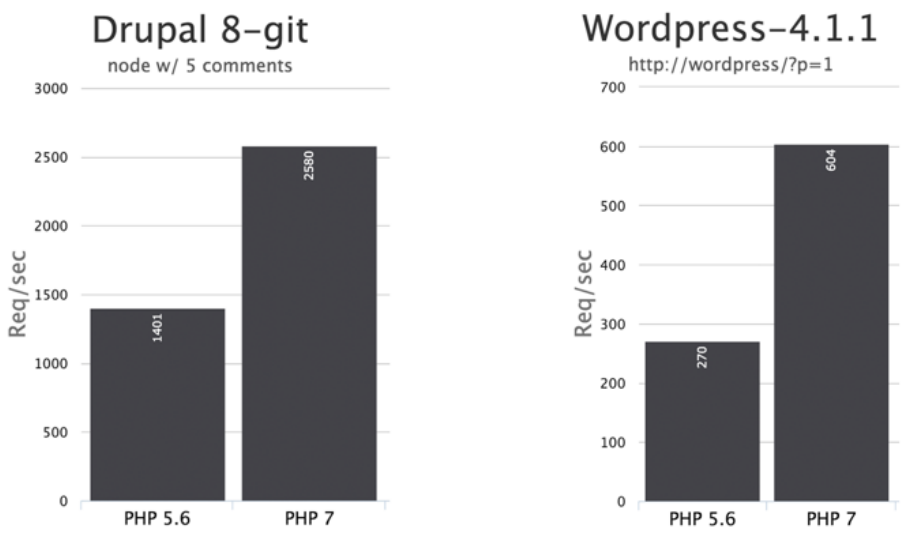

Figura 3.6: Diferencias de rendimiento entre PHP 5.6 y PHP 7

MySQL

Cuando hablamos de bases de datos relacionales, hay varias opciones que elegir, sin embargo a la hora de buscar tecnologías he decidido tener en cuenta MySQL como base de datos Structured Query Language (SQL) por ser un sistema de gestión de bases de datos relacional con una licencia dual (pública general y comercial) y que además es considerada como la base de datos de código abierto más popular del mundo. Por si no fuera poco, utilizada en conjunto con phpMyAdmin se consigue una muy buena gestión visual de la misma.

Para este proyecto no se plantea a la base de datos como unas de las partes principales de la aplicación ya que la información se obtiene en gran parte a través de una API externa, sin embargo, su uso es indispensable para almacenar cierto tipo de información y para poder crear alguna que otra rutina de automatización.

3.7.3. El desarrollador web completo

Últimamente cada vez está más demandada la figura del Full Stack Developer [34], el cuál es un desarrollador web completo, el cual debe de conocer tanto la parte front-end como la parte back-end y saber desenvolverse en los diferentes sistemas operativos.

Según la revista Business Insider en su versión norteamericana, este ha sido el perfil más demandado en el mundo laboral durante el año 2018, ya que al ser capaz de establecer estrategias en todas y cada una de las partes del proyecto, es una persona todoterreno con mucho conocimiento técnico e imprescindible en las empresas tecnológicas.

De igual forma, se valoran conocimientos en posicionamiento Search Engine Optimization (SEO) ,

servicios en la nube, sistemas de control de versiones, seguridad y análisis de estadísticas.

3.8. Conclusiones del análisis

Tras analizar todo lo descrito anteriormente, la creación de una aplicación en la que poder gestionar varias cuentas bancarias desde un mismo lugar y de una manera fácil es una buena opción y viable para el día de hoy con la implantación de PSD2 .

A la hora de elaborar este TFG hay que tener en cuenta el aspecto económico al ser un desarrollo a nivel personal al que no se le va a sacar beneficio económico y por lo tanto diversas opciones de pago deben ser descartadas por su coste, debiendo buscar una fórmula que cumpla los objetivos y sea accesible. En mi caso en cuanto a la elección de la forma de obtener los datos, he considerado optar por la API de Salt Edge al ser la opción que más se ajusta a las características concretas del proyecto, que se resumen en la tabla 3.1

	Afterbanks	**Eurobits**	**Salt Edge**	**Plaid**	**Tink**
Bancos españoles	Todos	Todos	Muchos	Nulos	Pocos
Funcionalidad API	Media	Desconocido	Muy alta	Media	Media
Dificultad integración	Muy baja	Desconocido	Media	Baja	Baja
Calidad documentación	Baja	Desconocido	Alta	Alta	Alta
Idioma interfaz	Español	Español	Multilenguaje	Inglés	Inglés
Precio	De pago	De pago	Plan gratuito	Plan gratuito	Plan gratuito

Tabla 3.1: Tabla comparativa de proveedores AISPs

Tras analizar su documentación y aunque es algo extensa debido a la gran posibilidad de acciones y opciones de configuración que se pueden realizar, es totalmente viable su combinación con las tecnologías front-end y back-end analizadas, que me permiten ser capaz de desarrollar una web reactiva y muy visual de forma que su manejo esté al alcance de todos.

En la parte front-end he optado por usar Vue.js por su gran futuro gracias a la comunidad de usuarios que tiene y la mayor sencillez a la hora de realizar ciertas tareas, que junto a Bootstrap con un tema que me ha gustado mucho llamado Yeti (funcionando mediante compilación SASS para el estilo), son una buena combinación.

A nivel back-end tras analizar una serie de aspectos para conectar una base de datos MySQL con la API externa, la mejor forma de realizar el tratamiento es con PHP Además, al utilizar PHP es muy sencillo hacer la compatibilidad con una base de datos PostgreSQL con el uso de la librería PHP Data Object (PDO) [35] para hacer las conexiones, resultando al final una ayuda para lograr una mayor universalidad del proyecto.

3.8.1. Tratamiento de la información

La aplicación GlobalBank es la encargada de tratar la información bancaria del usuario final, sin embargo la obtención de datos es un aspecto externo por parte de Salt Edge, el proveedor de información bancaria. Para crear un banco nuevo y al utilizar una licencia de uso simplificada, con el fin de garantizar la seguridad, los datos bancarios a transmitir (usuarios y contraseñas) son tratadas por su sistema que ha sido auditado y cuenta con las máximas garantías de seguridad y luego mi aplicación accede a los datos por la API con el correspondiente token generado.

En un primer momento me planteé el hecho de guardar en la base de datos local todos los datos de saldos y transacciones y que esa información se fuera actualizando a diario o a discreción del usuario. Sin embargo, teniendo en cuenta que la API responde muy rápido a la información que ya está almacenada en el proveedor y que la operación no se va a estar ejecutando constantemente, se puede confiar en la disponibilidad y garantía de utilizar únicamente la API externa para obtener estos datos. Entonces, la base de datos local se utilizará principalmente para tener todo lo requerido para hacer las llamadas al proveedor o para gestionar la visualización propia: tokens, códigos de banco, bancos admitidos con sus correspondientes logos y las configuraciones del usuario.

Esta forma de tratar los datos gracias las tecnologías front-end y back-end decididas, se complementan muy bien y son capaces de ofrecer un muy buen rendimiento de la aplicación y que la experiencia por parte del usuario sea muy buena y acorde a lo esperado.

4

Diseño y desarrollo

4.1. Introducción

En este apartado se van a exponer los aspectos más destacables en cuanto al diseño de la aplicación. Se abordarán las implementaciones más características o curiosas de las tecnologías utilizadas.

4.2. Back-end

Para la parte de Back-end hay que destacar el uso de PHP y una base de datos MySQL que almacena cierto tipo de información como los token de las cuentas bancarias agregadas y la información de los bancos compatibles junto a sus imágenes para poder interactuar con ello en la aplicación.

4.2.1. Conexiones con la API

En primer lugar para entender bien cómo se deberían hacer las llamadas a la API y estructurar de forma interna el proyecto, me he estudiado la documentación del proveedor Salt Edge que decidí utilizar. Para hacer las pruebas correspondientes y analizar el JSON, recurrí a la herramienta Postman.

Una vez verificadas las llamadas y planteado cómo estructurar la aplicación, hay que realizar esas mismas peticiones a través del código PHP de mi aplicación, para lo cual es muy útil la librería cURL [36] que permite enviar, entre otras, las peticiones GET, POST, PUT y DELETE requeridas y obtener en formato JSON las respuestas para ser fácilmente interpretadas más adelante.

4.2.2. Integración de la API

Como ya comenté a la hora de analizar el proveedor, las credenciales de los usuarios deben ser introducidas a través de la plataforma de conexión de Salt Edge, la cual recibe el nombre de Salt Edge Connect. En cuanto a las posibilidades, se podía optar por una redirección a su plataforma en la que el

usuario eligiese el banco y rellenase la información o podía jugar con los parámetros de las llamadas a la API y ser yo quien en un primer momento mostraba al usuario los bancos compatibles a través de mi web, para encaminarles directamente hacia el inicio de sesión de ese banco en concreto. Opté por la segunda opción con el fin de poder facilitar su uso y mostrar de manera más visual las distintas opciones disponibles.

A medida que el usuario va interactuando con Salt Edge Connect (figura 4.1), mi aplicación GlobalBank va recibiendo mediante POST a una dirección dada, llamadas de retorno con información del proceso, las cuales me encargo de analizar y tratar.

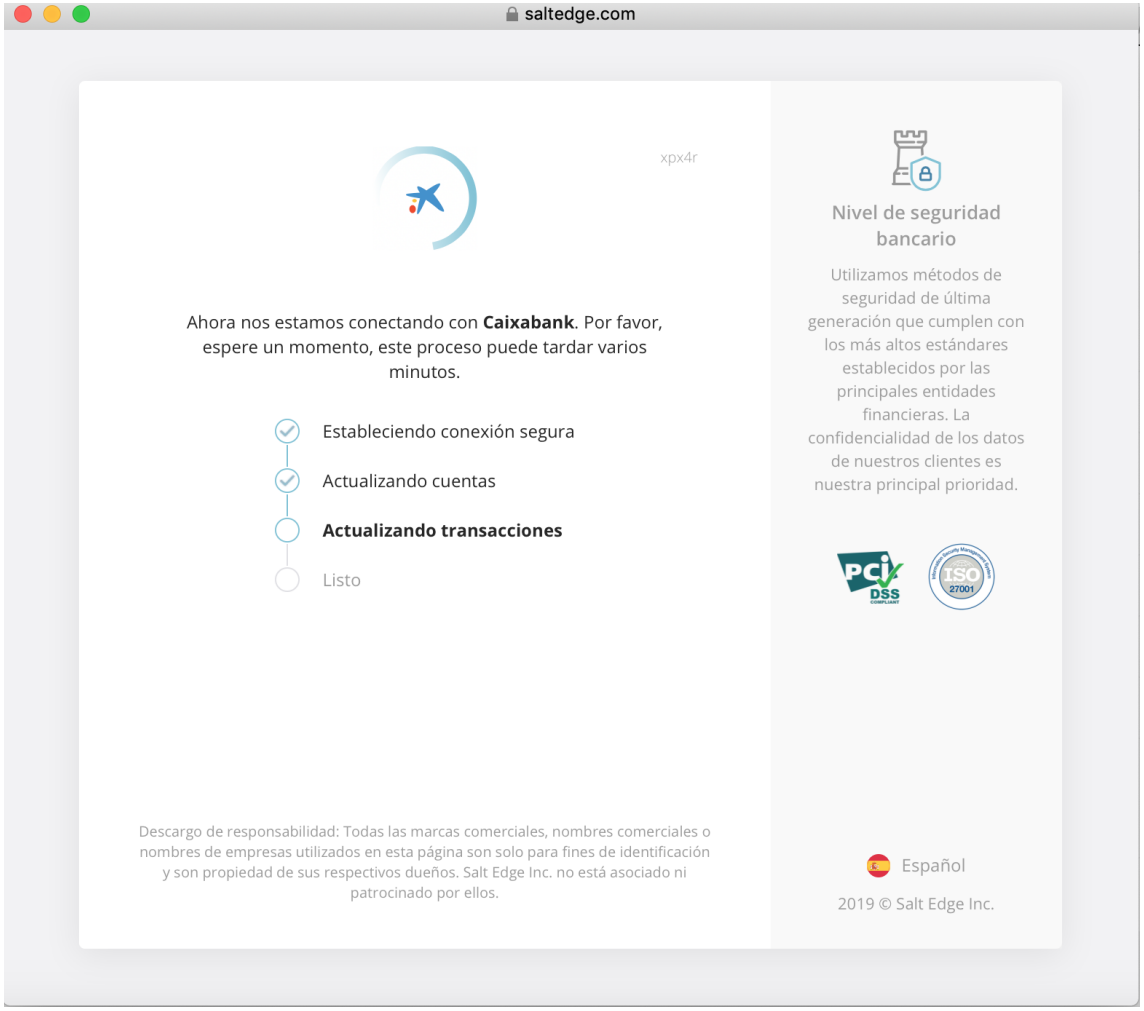

Figura 4.1: Proceso de agregación de nuevo banco en Salt Edge Connect

Una vez establecida la lógica de la aplicación a nivel back-end, desde phpMyAdmin puedo ver de forma rápida la información de la base de datos, además de poder hacer cambios o consultas de una forma cómoda.

Una vez que es posible obtener todos los datos aplicando la correspondiente lógica a las llamadas según la acción del usuario, el trabajo de mostrado de los mismos en la web, queda en manos de los mecanismos front-end.

4.2.3. Mejoras de la API

Con respecto a las posibilidades que ofrece la API de Salt Edge, una de ellas es que en la versión personal gratuita, ellos no se encargan de la actualización automática de la información bancaria, sino que delegan la tarea a la aplicación.

De esta forma, el comportamiento esperado es que el usuario final debe actualizar de forma manual cada banco realizando la correspondiente petición. Con el fin de simplificar este proceso y que el usuario no tenga que estar pendiente de actualizar a diario sus cuentas, he implementado una opción en la que se puede elegir libremente si desea que un determinado banco se actualice o no de forma automática.

Para lograr este comportamiento, he creado un script propio que se encarga de enviar de forma masiva las peticiones de actualización de las cuentas bancarias deseadas. Para lograr que se realice la tarea a diario, tan solo es necesario programar a nivel de servidor la ejecución del script con una cierta periodicidad, tal y como se especifica en la Guía del Programador (Anexo A).

4.3. Front-end

Para la parte de Back-end hay que destacar el uso de Vue.js junto a Bootstrap, y además de haber recurrido a ciertas librerías adicionales de JavaScript desarrolladas por terceros para implementar alguna que otra funcionalidad que da un gran valor a la página.

4.3.1. Bootstrap

A la hora de estructurar la visualización de la página web en cuanto a términos de estética, Bootstrap ha sido clave por su grandes posibilidades y elementos disponibles. La división por cuadrículas de Bootstrap permite obtener unas interfaces muy limpias y ordenadas

En mi caso en concreto, he utilizado el tema Yeti por el diseño minimalista de las tablas y su gama de colores, que al no ser tonos muy fuertes como si se encontraba en otros temas, se consigue una satisfacción de uso mucho mayor.

4.3.2. Vue.js

Para aprender cómo utilizar este framework que era totalmente nuevo para mi, he revisado el curso gratuito ”Aprende Vue2 y Firebase paso a paso” [37] en el que se explica desde el principio qué es Vue.js y sus posibilidades reactivas. Desde cosas tan sencillas como los ciclos de vida de los componentes (figura 4.2) hasta su posible despliegue en servidores para el caso de amplios proyectos.

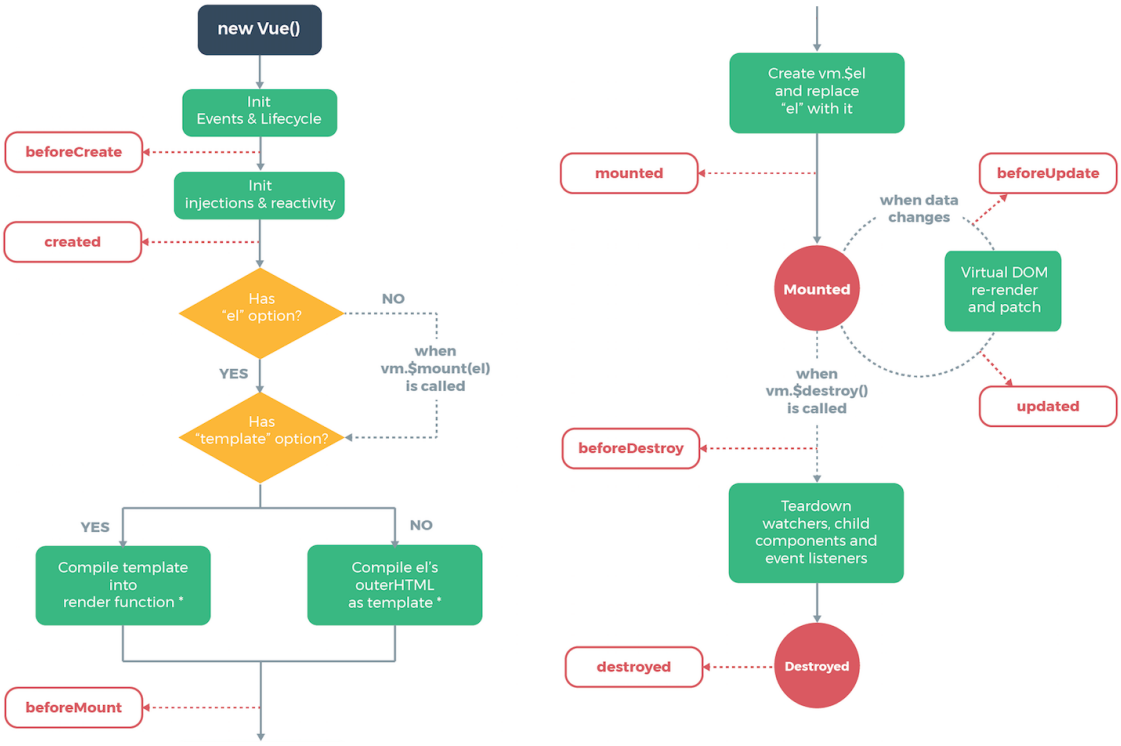

Figura 4.2: Ciclo de vida de un componente Vue.js

Para visualizar la información dentro de la aplicación GlobalBank, se han creado componentes Vue.js con sus características (variables y acciones). Dentro de la parte de acciones, pueden ser los correspondientes GET para obtener las respuestas de la API , la realización de diferentes acciones matemáticas, la aplicación de filtros de visualización para convertir datos en texto más amigable al ojo humano o la observación de cambios en variables a tiempo real, entre otras cosas.

En la figura 4.3(a) se puede visualizar un ejemplo de componente, y las diferentes acciones según el estado del mismo.

Una de las ventajas de Vue.js, es que aunque tiene una curva de aprendizaje rápida, el hecho de aprender bien el lenguaje y el buen uso de las promesas, te permite hacer en un trozo muy pequeño de código, algo que resultaría más complejo de otra manera y asegurando una compatibilidad universal por ser una base JavaScript.

A la hora de interactuar el modelo con la vista, se consiguen unos códigos HTML muy limpios y compactos propios de las pequeñas aplicaciones, pero con unas funcionalidades muy amplias por detrás. En la figura 4.3(b) se puede ver un ejemplo de la página de transacciones y el uso de varias sintaxis de condiciones y bucles en combinación con elementos de interacción con el usuario a través de un deslizador y el mostrado de la información con la interpolación de los corchetes.

```
      created: function(){
        NProgress.start(); /* Iniciar animación de cargando */
        this.get_saldos(); /* Llamar a función para obtener saldos */
      },
      computed: {
        /* Obtención del total de saldo de todos los bancos al computar el neto de cada uno */
        totalGlobal() {
          return this.saldos.map(({data}) => data).flat().reduce((a, b) => a + b.balance, 0);
        },
      },
      methods:{
        /* Obtención del JSON con la información de cuentas y saldos */
        get_saldos: function(){
          fetch("./mysqlapi.php?action=saldosapi")
          .then(response=>response.json())
          .then(json=>{this.saldos=json.saldos});
        },
        /* Obtención del saldo neto total contabilizando todos los productos contratados */
        get_totalBanco: function(bank){
          return bank.data.reduce((a, b) => a + b.balance, 0)
        },
      },
      watch: {
        /* Observador de cambio de la variable saldos */
        saldos: function() {
          NProgress.done(); /* Finalizar animación de cargando */
        }
      },
      filters: {
        /* Filtro para convertir una fecha dada al formato dd/mm/yyyy hh:mm */
        moment: function (date) {
          return moment(date).format('DD/MM/YYYY HH:mm');
        }
```

(a) Ejemplo de componente Vue.js

```
<div v-if="item_length == 0 || numtransacciones == 0">No hay ninguna transacción para mostrar en esta cuenta</div>
<div v-if="numtransacciones == 1">A continuación se puede visualizar la última transacción del producto bancario escogido: </div
<div v-else-if="numtransacciones == 0 || item_length == 0"></div>
<div v-else>A continuación se pueden visualizar las {{numtransacciones}} últimas transacciones del producto bancario escogido: <

<table v-if="item_length > 0 && numtransacciones > 0" class="table table-hover">
<thead>
  <tr>
    <th style="width: 30%">Fecha</th>
    <th style="width: 60%">Descripción</th>
    <th style="width: 10%">Importe</th>
  </tr>
</thead>
<tbody v-for="transaccion in transacciones.slice(0,numtransacciones)">

  <tr>
    <td>{{transaccion.made_on}}</td>
    <td>{{transaccion.description}}</td>
    <td v-if="transaccion.amount < 0" id="negativo">{{transaccion.amount.toLocaleString(undefined, {minimumFractionDigits: 2,max
    <td v-else id="positivo">{{transaccion.amount.toLocaleString(undefined, {minimumFractionDigits: 2,maximumFractionDigits: 2}}
  </tr>
</tbody>
</table>
```

(b) Ejemplo de fichero HTML de una página Vue.js

Figura 4.3: Ejemplos de utilización del framework Vue.js

Moment.js

La librería Moment.js es muy completa para tratar y manipular las fechas en distintos formatos de representación. Esto en combinación con la posibilidad de crear filtros en Vue.js facilita la interpretación y presentación de los datos a los usuarios que hagan uso de la aplicación.

NProgress.js

El uso de NProgress.js permite mostrar de forma visual el estado de carga de la página web, con la combinación de una barra en la parte superior de la página y un círculo que gira en mitad de la pantalla, de forma que mientras se gestionan las llamadas externas y se analiza la información, el usuario sea consciente de que la página no ha terminado de cargar y que pronto mostrará la información solicitada. Esta librería consta de un JavaScript y un CSS con unos aspectos de visualización en concreto.

En este caso me he tomado el trabajo de hacer una serie de personalizaciones sobre la librería para adaptarlo al resultado final que buscaba obtener.

Charts.js

Para representar dentro de mi aplicación unas gráficas de una forma bastante atractiva, he recurrido a la librería Charts.js, que aunque la documentación es algo confusa en ciertos aspectos y no tienen muchos ejemplos, es una extensión que tiene una gran cantidad de posibilidades.

Para mi caso en concreto, he querido mostrar la información de gastos e ingresos de los últimos 30 días de cada cuenta bancaria en cuestión, pudiendo ver a simple vista cuál es la tendencia de dicho producto. Un ejemplo de puede visualizar en la figura 4.4.

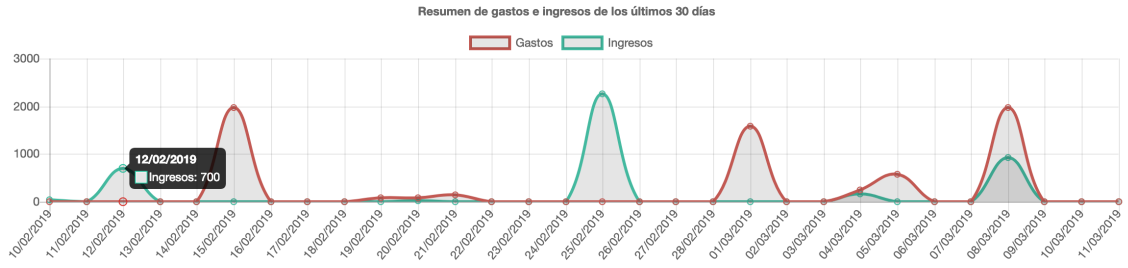

Figura 4.4: Ejemplo de gráfico con Chart.js en base a la información de una cuenta bancaria

4.4. Interfaz de usuario

Una vez completado el desarrollo, y habiendo logrado una aplicación final totalmente funcional, las imágenes de la interfaz resultante se pueden visualizar en el apéndice B.

5 Integración, pruebas y resultados

5.1. Introducción

Dentro de un proyecto de software es necesario dedicar un tiempo a una fase de realización de pruebas con el fin de garantizar una calidad esperada y un correcto funcionamiento de todos los módulos. Aunque el hecho de realizar estas pruebas conlleva un mayor tiempo de desarrollo hasta que el producto se pone a disposición de los usuarios finales, es algo necesario para obtener un buen acabado que cumpla con los requisitos establecidos y no tenga que acabar siendo retirado debido a los problemas que se vayan encontrando.

Dentro de este apartado se analizan las diferentes pruebas realizadas desde el punto de funcionalidad, usabilidad, compatabilidad y sobre el entorno final de despliegue.

5.2. Funcionalidad

Se han realizado pruebas y han resultado ser todo un éxito, cumpliendo todos los requisitos establecidos en la fase de análisis.

A la hora de ir añadiendo las funcionalidades se ha ido haciendo de manera incremental desde los aspectos más generales hasta los más específicos, haciendo las correspondientes pruebas a la finalización de cada uno de estos ciclos. Cualquier problema detectado o característica que tenía ser modificada por cualquier motivo, era realizado antes de pasar a la siguiente fase con el fin de no arrastrar algo que ya se pensaba cambiar.

Dentro de estas comprobaciones, se han analizado casuísticas varias con diferentes posibilidades con el fin de verificar que se abarcan todos los casos posibles.

Por un lado, tenemos las pruebas de caja blanca donde sabiendo la ingeniería del código se busca el encontrar posibles errores que produzcan un comportamiento inesperado. Para ello, se han usado datos JSON reales y también simulados rellenando o dejando en blanco una serie de campos según las posibles variaciones entre cada banco, con el fin de verificar que aún con ligeros cambios en su

estructura la información se muestra como es debido. En este punto es de vital importancia tratar los casos con datos corruptos o malformados, con campos vitales vacíos y con casos de un solo objeto dentro de un bucle que espera múltiples elementos.

Entre los casos que más problemas me dieron fue al mostrar determinados datos como los nombres de los productos bancarios o los saldos dispuestos y disponibles de las tarjetas de crédito.

Para el problema de los nombres de los productos, hay algunos bancos que por la API no devuelven el nombre sino únicamente el número, o incluso he tenido algún caso en el que el nombre viene fijado en otro campo extra que en teoría no está previsto para ese fin, por lo que hay que analizar todo el JSON para buscar dicha existencia o si no la hay, para mostrar un texto genérico que además debe ser acorde al tipo de producto bancario en cuestión.

A la hora de trabajar con las tarjetas de crédito los problemas han sido mayores ya que en el campo de saldo, unos te mostraban el saldo dispuesto todavía no liquidado mientras que otros mostraban un valor de 0€, por lo que podía ser confundido con una tarjeta de débito, que también aparecen como producto dentro de la aplicación para poder ver los movimientos, aunque éstas no tiene un saldo como tal. Para decidir si se trata de una tarjeta de débito o crédito y mostrar el texto apropiado o el posible saldo dispuesto cuando la respuesta del banco no diferencia entre ambas en el campo de saldo, he tenido que analizar entre las propiedades para ver si hay algún campo que informase sobre un total mensual concedido (lo que ya nos indicaría que efectivamente si es de crédito y no de débito) además de otro campo con la información total disponible en este periodo de facturación con el fin de hacer manualmente la operación para deducir cuál es la deuda real del cliente (saldo dispuesto) y poder mostrarlo en la aplicación para que el estado de cuentas global sea acorde a su verdadera situación financiera.

Al estar trabajando con JavaScript y aunque existe una consola donde poder detectar errores, es un lenguaje que no muestra los fallos en la interfaz sino que se comporta de manera extraña y es muy difícil de depurar si no se hace caso por caso en pequeñas unidades.

Dejando de lado las pruebas de caja blanca, por el otro lado, cuando el proyecto se encontraba ya en una fase más avanzada, se han hecho pruebas más generales de caja negra en las que se conoce la entrada y la salida y no es necesario centrarse en la lógica interna. Se espera que tras estas acciones, la aplicación funcione como debe y que las interacciones entre los distintos módulos de la aplicación hayan producido el comportamiento esperado.

5.3. Usabilidad

El hecho de la usabilidad es una de las fases más críticas del proceso ya que la finalidad es que todo sea intuitivo para el usuario final y que no tenga dudas en ningún momento de las acciones que

va a realizar o cómo debe hacer otras.

Es una aplicación simple en el aspecto que vale más que sea sencilla y fácil de utilizar según las características que en un primer momento se desearon, a añadir mucha funcionalidad inútil que realmente no se le va a sacar partido en el día a día y que lo único que hacen es afectar a la sencillez, pero sobre todo el rendimiento de la aplicación.

Para medir la calidad de la experiencia del usuario, he recurrido a varios beta testers con diferentes niveles de manejo informático a los que se les ha pedido que realicen una serie de acciones y expresen su experiencia, la claridad para realizar la acción, si han echado algo de menos y un comentario general.

5.4. Compabilidad universal

Con las tecnologías front-end que se han utilizado, en este punto no ha habido ningún problema. En un momento si que hubo que realizar un pequeño ajuste de unos botones para su correcta visualización en pantallas pequeñas, pero al ser una aplicación web desde el inicio no ha habido muchos inconvenientes.

Estas tecnologías modernas es cierto que requieren que los navegadores estén actualizados y no es compatible con Internet Explorer por ejemplo (si es compatible con Microsoft Edge), sin embargo dicha versión es ya muy antigua y su uso es muy reducido. De igual forma, entre los requisitos no funcionales no se especifica esta compatibilidad y el requisito con los navegadores mínimos indicados en sus versiones más modernas no ha causado ningún problema.

Se ha probado en los 3 sistemas operativos grandes para ordenadores (Windows, Linux y Mac OS) y en los 4 navegadores especificados (Google Chrome, Mozilla Firefox, Microsoft Edge y Apple Safari) sin ningún inconveniente. Por la parte móvil, las pruebas han sido sobre los 2 sistemas operativos principales (Android y iOS) con sus navegadores nativos o instalando Google Chrome en caso de que no viniera de fábrica y los resultados han sido positivos cuando se trabajaba en versiones más o menos actualizadas.

5.5. Servidores de pruebas

Para desplegar mi aplicación y verificar su correcto funcionamiento he utilizado tanto un servidor remoto como uno local, superando las pruebas sin ningún inconveniente.

Para las pruebas en remoto, he utilizado un servidor de web hosting online basado en una arquitectura CentOS sobre la que está instalado web hosting de tal forma que he obtenido un fácil acceso al proceso de creación de base de datos MySQL y manipulación de la misma mediante phpMyAdmin. He

instalado correctamente la aplicación a través del asistente y el funcionamiento ha sido el esperado, accediendo correctamente desde mi dominio web. La parte de automatización de la actualización de saldos es trivial al poder configurar desde el propio panel del alojamiento un cronjob que ejecuta el fichero de actualización todas las mañanas.

Para la parte local, he hecho pruebas en los 3 sistemas operativos (Windows, Mac OS y Linux), levantando un servicio Apache y otro servicio MySQL, de forma que tras realizar las correspondientes configuraciones de rutas y usuarios, he logrado a través del localhost un acceso completo y funcional, incluyendo la correcta ejecución del instalador. La parte de automatización de la actualización de saldos es algo más complejo en este caso al tener que recurrir a procedimientos de programación de tareas programadas del propio sistema operativo que son ajenas a los servicios webs, ya que va más enfocado a automatizaciones del propio sistema operativo.

6

Conclusiones finales y a futuro

6.1. Conclusiones finales

En este proyecto y a pesar de la limitación en cuanto a los recursos económicos, el resultado final ha sido la creación de una aplicación totalmente funcional para un ámbito personal que permite el acceso a información bancaria real y actualizada de distintos bancos, unificándolo desde un solo lugar.

La aplicación es capaz recopilar los datos sobre productos financieros contratados en 18 entidades bancarias españolas, ofreciendo un resumen global del patrimonio y permitiendo la consulta de saldos y transacciones de cada uno de estos productos.

El coste intelectual ha sido alto al requerir hacer un análisis exhaustivo de las normativas, comunicarme con diversas empresas por correo electrónico y teléfono, estudiar las documentaciones de distintos proveedores, aprender nuevos frameworks o lenguajes de programación punteros y familiarizarme con diversas herramientas útiles de gestión.

El tiempo dedicado a la evaluación inicial para el acceso a la información bancaria ha sido muy importante ya que me ha permitido descartar opciones que en un primer momento parecían interesantes y necesarias para el futuro desarrollo y que si hubiera optado por ellas me hubieran obligado a rehacer y adaptar una parte del trabajo porque la funcionalidad que ofrecían era limitada.

Haber aprendido el framework Vue.js ha sido clave para conseguir una funcionalidad reactiva y compatible con todos los navegadores actuales, exprimiendo las diferentes posibilidades que ofrece el lenguaje logrando obtener una web dinámica, con tecnología moderna y robusta.

La apuesta por Bootstrap me ha permitido que la parte gráfica sea intuitiva con unos estilos llamativos y con elementos adaptados a todo tipo de pantallas.

A destacar de mi trabajo, considero que la funcionalidad lograda permite prescindir de las aplicaciones oficiales de los bancos al ser capaz de ofrecer la misma información de una forma organizada en menos pasos y el permitir mediante el uso de colores distinguir de forma clara la información relevante de cada producto.

Asimismo, la aplicación tiene alguna limitación como el hecho de que ha sido adaptada para un uso personal y no se ha tenido en cuenta un acceso multiusuario a la misma. Cada instancia de la aplicación queda asociada a un usuario en concreto.

Otro aspecto que puede generar algún inconveniente es que para la actualización automática de los datos es necesario configurar una tarea programada para ejecutar el script de actualización, no obstante, dicha solución casera es una mejora con respecto a lo que ofrece de serie el proveedor que obligaría a actualizar de forma individual cada banco con su correspondiente botón.

En definitiva, el objetivo del proyecto se ha cumplido con una satisfacción alta.

6.2. Trabajo futuro

Este proyecto consistía en el desarrollo de una aplicación de control de cuentas bancarias multi entidad, desde la que poder ver los saldos y movimientos de diferentes cuentas bancarias.

Actualmente se están utilizando las herramientas de una página web proveedora de información extranjera, pero en los próximos meses vamos a ver mucho movimiento en este aspecto y se podría plantear la opción de hacer una migración a la nueva plataforma que está preparando Redsys, más adaptada al mercado nacional y que con la nueva normativa será capaz de obtener todavía muchos más datos de cada cuenta bancaria.

Otro de los aspectos interesantes a tener en cuenta cuando lo anterior sea una realidad, es la de hacer una ampliación de la funcionalidad y que la web no solo sea para consultar información, sino que también se puedan hacer operaciones con las cuentas, como puede ser una transferencia sin necesidad de abandonar la aplicación.

Incluso en un proyecto mucho más ambicioso cuando todos los bancos estén adaptados a la normativa y su acceso a los datos sea aún más sencillo, se podría llegar a plantear el hecho de que la aplicación sea multi usuario.

Bibliografía

[1] "El auge de las fintech en España." [En línea]. Disponible en: Stock Crowd IN. [Accedido: 17-May-2019].

[2] "El auge de las fintech desde el punto de vista de los bancos." [En línea]. Disponible en: Selfbank. [Accedido: 17-May-2019].

[3] "30 años de la firma: Tratado de Adhesión de España a la UE." [En línea]. Disponible en: Parlamento Europeo. [Accedido: 17-May-2019].

[4] "Euro." [En línea]. Disponible en: Wikipedia. [Accedido: 17-May-2019].

[5] "Fintonic." [En línea]. Disponible en: Fintonic. [Accedido: 17-May-2019].

[6] "Klarna." [En línea]. Disponible en: Klarna. [Accedido: 17-May-2019].

[7] "Trustly." [En línea]. Disponible en: Trustly. [Accedido: 17-May-2019].

[8] "Instantor." [En línea]. Disponible en: Instantor. [Accedido: 17-May-2019].

[9] "La PSD: la normativa que abrió camino a la PSD2." [En línea]. Disponible en: BBVAOpen4u. [Accedido: 17-May-2019].

[10] "¿Qué es SEPA y qué es el IBAN?." [En línea]. Disponible en: En Naranja. [Accedido: 17-May-2019].

[11] "La banca inicia la cuenta atrás para la aplicación de la PSD2." [En línea]. Disponible en: BBVA. [Accedido: 17-May-2019].

[12] "Banca abierta, el despertador de las entidades tradicionales." [En línea]. Disponible en: Hablemos de empresas. [Accedido: 17-May-2019].

[13] "PSD2: más seguridad y protección para el consumidor." [En línea]. Disponible en: Xeridia. [Accedido: 17-May-2019].

[14] "Screen scraping: Todo lo que debes saber sobre este término." [En línea]. Disponible en: Inespay. [Accedido: 17-May-2019].

[15] "Screen scraping, el caballo de batalla de PSD2." [En línea]. Disponible en: Revolution Banking. [Accedido: 17-May-2019].

[16] "Todo lo que hay que saber de la PSD2." [En línea]. Disponible en: BBVA. [Accedido: 17-May-2019].

[17] "BBVA API Market." [En línea]. Disponible en: BBVA. [Accedido: 17-May-2019].

[18] "Las APIs más conocidas de agregación de datos bancarios." [En línea]. Disponible en: BBVAOpen4u. [Accedido: 17-May-2019].

[19] "ING Developer Portal." [En línea]. Disponible en: ING. [Accedido: 17-May-2019].

[20] "Afterbanks, la API para acceder a los bancos." [En línea]. Disponible en: Afterbanks. [Accedido: 17-May-2019].

[21] "Eurobits, tu partner PSD2." [En línea]. Disponible en: Eurobits. [Accedido: 17-May-2019].

[22] “Webinar PSD2: agregar o ser agregado.” [En línea]. Disponible en: Youtube. [Accedido: 17-May-2019].

[23] “Agregar o ser agregado: la importancia de moverse rápido.” [En línea]. Disponible en: Eurobits Blog. [Accedido: 17-May-2019].

[24] “Todos tus bancos son bienvenidos a la app de BBVA con el nuevo servicio de agregación.” [En línea]. Disponible en: Youtube. [Accedido: 17-May-2019].

[25] “Salt Edge Docs: Account Information.” [En línea]. Disponible en: Salt Edge. [Accedido: 17-May-2019].

[26] “Plaid: The easiest way for users to connect their bank accounts to an app.” [En línea]. Disponible en: Plaid. [Accedido: 17-May-2019].

[27] “We are data-driven banking.” [En línea]. Disponible en: Tink. [Accedido: 17-May-2019].

[28] “Bootstrap: The most popular HTML, CSS, and JS library in the world.” [En línea]. Disponible en: Bootstrap. [Accedido: 17-May-2019].

[29] “React – A JavaScript library for building user interfaces.” [En línea]. Disponible en: ReactJS. [Accedido: 17-May-2019].

[30] “Vue.js.” [En línea]. Disponible en: Vue.js. [Accedido: 17-May-2019].

[31] “¿Qué es Vue.js?.” [En línea]. Disponible en: Open Webinars. [Accedido: 17-May-2019].

[32] “Vue.js.” [En línea]. Disponible en: Node.js. [Accedido: 17-May-2019].

[33] “¿El desarrollo web con PHP es cosa de viejunos?.” [En línea]. Disponible en: Keepcoding. [Accedido: 17-May-2019].

[34] “¿Qué es Full Stack?.” [En línea]. Disponible en: Neoland. [Accedido: 17-May-2019].

[35] “Librería PDO.” [En línea]. Disponible en: PHP.net. [Accedido: 17-May-2019].

[36] “Librería cURL.” [En línea]. Disponible en: PHP.net. [Accedido: 17-May-2019].

[37] “Aprende Vue2 y Firebase paso a paso.” [En línea]. Disponible en: Teachable. [Accedido: 17-May-2019].

Definiciones

Apache Servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

beta tester Persona que prueba una versión previa a la final de un producto con el objetivo de descubrir errores y documentarlos de forma clara y precisa para que los programadores sean capaces de reproducirlos y solucionarlos antes del lanzamiento al público general.

CentOS Bifurcación a nivel binario de la distribución Linux Red Hat Enterprise Linux RHEL, compilado por voluntarios a partir del código fuente publicado por Red Hat, siendo la principal diferencia con este la eliminación de todas las referencias a las marcas y logos.

Cobol Lenguaje de programación universal creado en 1959 que podía ser usado en cualquier ordenador, y que estaba orientado principalmente a los negocios.

cronjob Administrador del sistema operativo Unix encargado de regular procesos en segundo plano al ejecutar tareas a intervalos regulares.

fintech Aglutina una industria de servicios financieros que utilizan la última tecnología existente para poder ofrecer productos y servicios innovadores.

Fortran Lenguaje de programación de alto nivel de propósito general, procedimental e imperativo, que está especialmente adaptado al cálculo numérico y la computación científica.

framework Estructura conceptual y tecnológica que mediante artefactos o módulos de software, sirven como base para el desarrollo de un producto de software.

JSON Formato de texto ligero para el intercambio de datos, siendo un subconjunto de la interpretación de objetos utilizada por Javascript.

localhost Nombre de dominio local reservado y vinculado a la IP 127.0.0.1 que tienen todos los ordenadores, routers y dispositivos independientemente de que dispongan o no de una tarjeta de red ethernet.

MySQL Sistema de gestión de base de datos relacional y de código abierto basado en lenguaje de consulta estructurado.

opensource Término que se utiliza para denominar a cierto tipo de software que se distribuye mediante una licencia que le permite al usuario final, si tiene los conocimientos necesarios, utilizar el código fuente del programa para estudiarlo, modificarlo y realizar mejoras en el mismo, pudiendo incluso hasta redistribuirlo.

phpMyAdmin Herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web.

PostgreSQL Sistema de gestión de bases de datos relacional orientado a objetos y de código abierto.

sandbox Entorno de pruebas separado del entorno de producción.

scoring Sistema de evaluación automática de solicitudes de operaciones de crédito.

screen scraping Screen scraping es el nombre en inglés de una técnica de programación que consiste en tomar una presentación de una información para, mediante ingeniería inversa, extraer los datos que dieron lugar a esa presentación.

web hosting Servicio que provee a los usuarios de Internet un sistema para poder almacenar contenido y que sea accesible vía web.

Acrónimos

ABE Autoridad Bancaria Europea.

AIS Account Information Service.

AISPs Account Information Service Providers.

API Application Programming Interface.

CERN European Organization for Nuclear Research.

CMS Content Management System.

CSS Cascading Style Sheets.

CSV Comma Separated Value.

FAQ Frequently Asked Questions.

HTML HyperText Markup Language.

HTTPS Hypertext Transfer Protocol Secure.

IBAN Internacional Bank Account Number.

LESS Leaner Style Sheets.

PDO PHP Data Object.

PHP Hypertext Preprocessor.

PIS Payment Initiation Service.

PISPs Payment Initiation Service Providers.

PSD Payment Services Directive.

PSD2 Revised Payment Services Directive.

SASS Syntactically Awesome Stylesheets.

SEO Search Engine Optimization.

SEPA Single Euro Payment Area.

SLA Service Level Agreement.

SQL Structured Query Language.

TFG Trabajo de Fin de Grado.

Apéndices

A

Guía del programador

La aplicación GlobalBank requiere para funcionar de una base de datos MySQL o PostgreSQL, y para completar las tareas de automatización, se debe programar para que un fichero PHP se ejecute una vez al día.

En primer lugar, se debe crear una base de datos MySQL o PostgreSQL y tener a mano los detalles de la misma: host, puerto, nombre de la base de datos, usuario y contraseña.

La instalación de la aplicación es muy sencilla, basta con copiar los archivos a un servidor web (ya sea un hosting online o la carpeta del servidor Apache a modo local).

La primera vez que se accede a la URL de la aplicación, aparece una página de instalación donde se solicitan los datos de conexión a la base de datos. Si todo va bien, se hace la automáticamente la creación de las tablas dentro de la base de datos. Si hay algún problema, se informa del error y se da la opción de corregirlo.

La contraseña por defecto para acceder a la aplicación es ”12345”. Para cambiar esta contraseña o si en algún momento es necesario actualizar los datos de conexión a la base de datos, se puede hacer en el fichero *config.php* (figura A.1).

Existe un fichero denominado *saltedgeconfig.php* (figura A.2) donde se configuran los parámetros de la aplicación propia que proporciona Saltedge al registraros (y en donde indicáis el dominio o la IP de donde está instalada la aplicación para que pueda recibir las llamadas de retorno).

Para configurar la automatización, se debe configurar una tarea en el equipo para que se ejecute el fichero *cronjob.php*. En los paneles de hosting online, como puede ser cPanel, hay una opción para crear esta tarea de una forma muy sencilla (figura A.3).

Al hacer la programación, es recomendable configurar la ejecución entre las 6:00 horas y las 23:00 horas ya que algunos bancos por la noche entran en un modo de mantenimiento y los servidores de dichos bancos son lentos atendiendo las solicitudes y en algunas ocasiones hay que hacer varios intentos hasta poder recibir la información y no mensajes de error. En mi caso lo he configurado que se ejecute todos los días, a las 7 de la mañana.

```php
<?php
define('DB_TYPE', 'MYSQL');
define('DB_HOST', 'localhost');
define('DB_PORT', '3306');
define('DB_NAME', 'javy97_tfg');
define('DB_USER', 'javy97_tfg');
define('DB_PASS', '3pqR1#f39703ia09');
define('APP_PASS', '12345');
?>

```

Figura A.1: Configuración de la base de datos y contraseña de la aplicación

```php
<?php
define('APPID', 'SHsKR16j2YIGG2YS80xRmiE24rW0waPcI6KnXFwN');
define('APPSECRET', 'OEpV6QvZAoVeQMYpW9OxEWPsns3UTGvv9Jh5pAUNODRtdDI0Cm');
define('CUSTOMERID', '2311165');
?>
```

Figura A.2: Configuración de los parámetros propios de Saltedge

Trabajos de cron actuales

Minuto	Hora	Día	Mes	Día de la semana	Comando	Acciones
0	7	*	*	*	/usr/local/bin/php /home/javy97/tfg.javiermarting.es/cronjob.php	Editar Borrar

Figura A.3: Configuración del cronjob en cPanel

B

Interfaz de usuario final

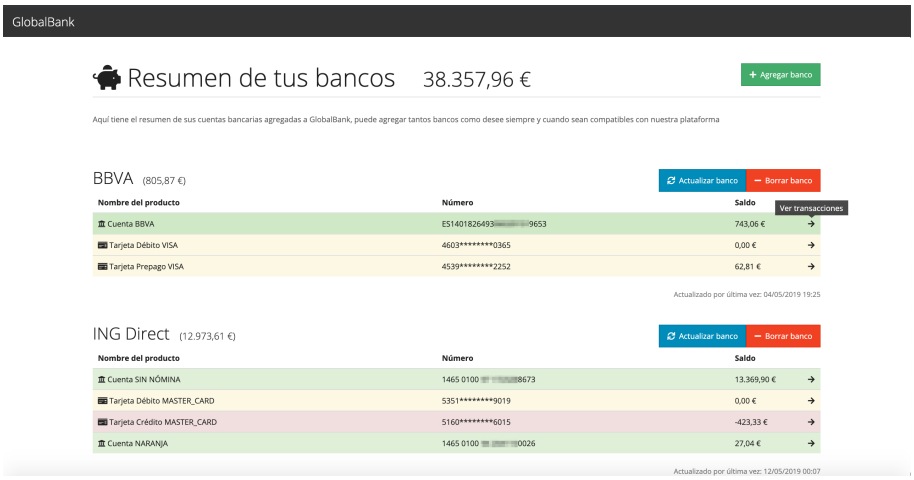

Figura B.1: Visualización de interfaz global en pantallas grandes

Figura B.2: Visualización de interfaz de movimientos en pantallas grandes

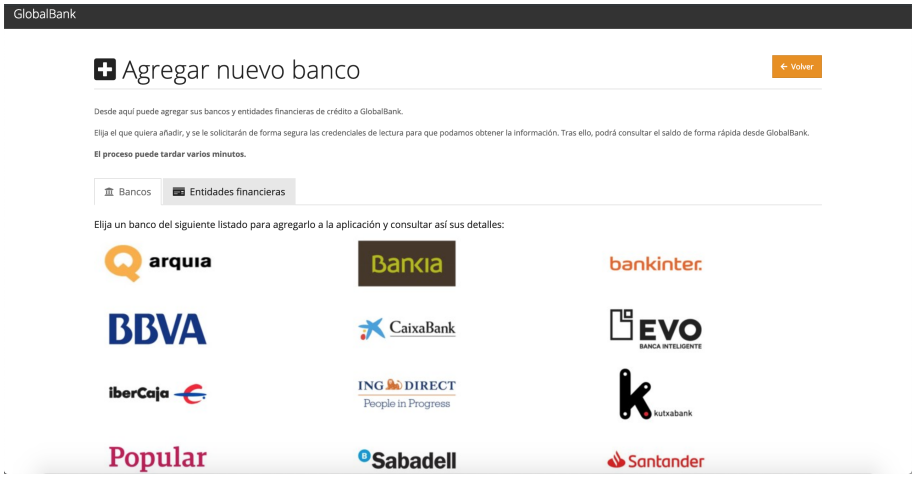

Figura B.3: Visualización de interfaz de agregar banco en pantallas grandes

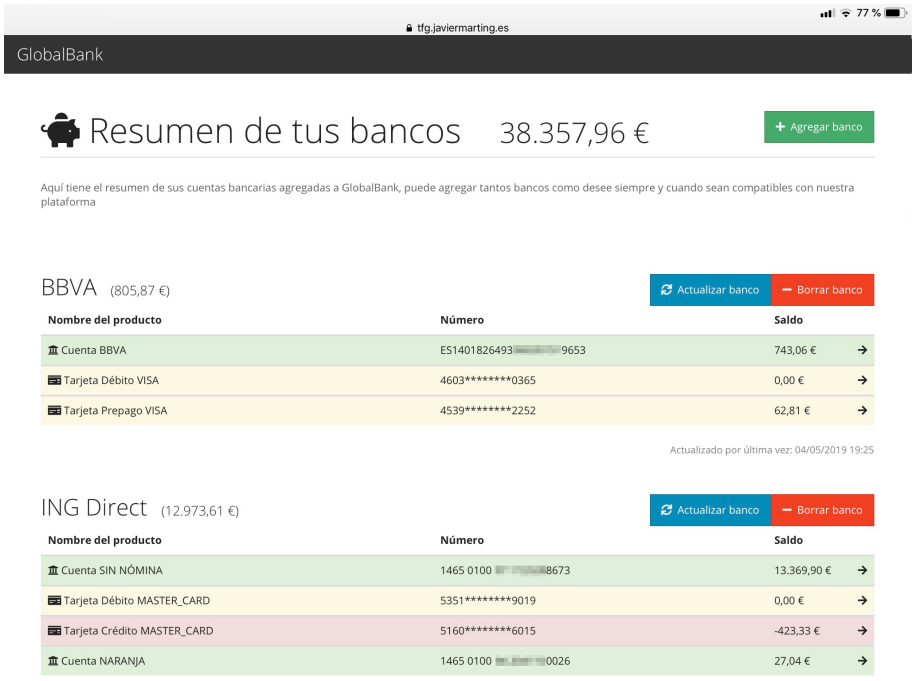

Figura B.4: Visualización de interfaz global en pantallas medianas

Figura B.5: Visualización de interfaz de movimientos en pantallas medianas

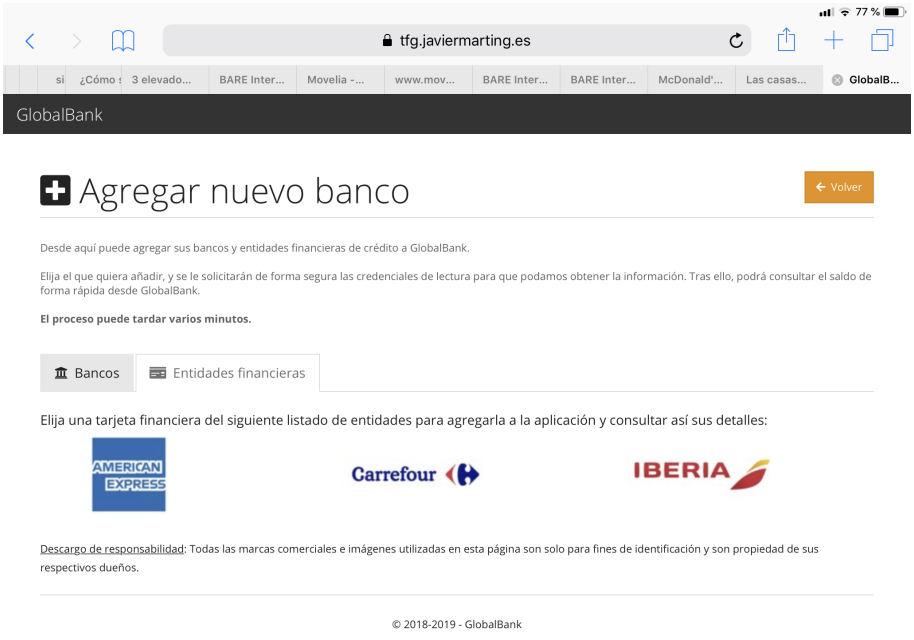

Figura B.6: Visualización de interfaz de agregar banco en pantallas medianas

Figura B.7: Visualización de interfaz global en pantallas pequeñas

Figura B.8: Visualización de interfaz de movimientos en pantallas pequeñas

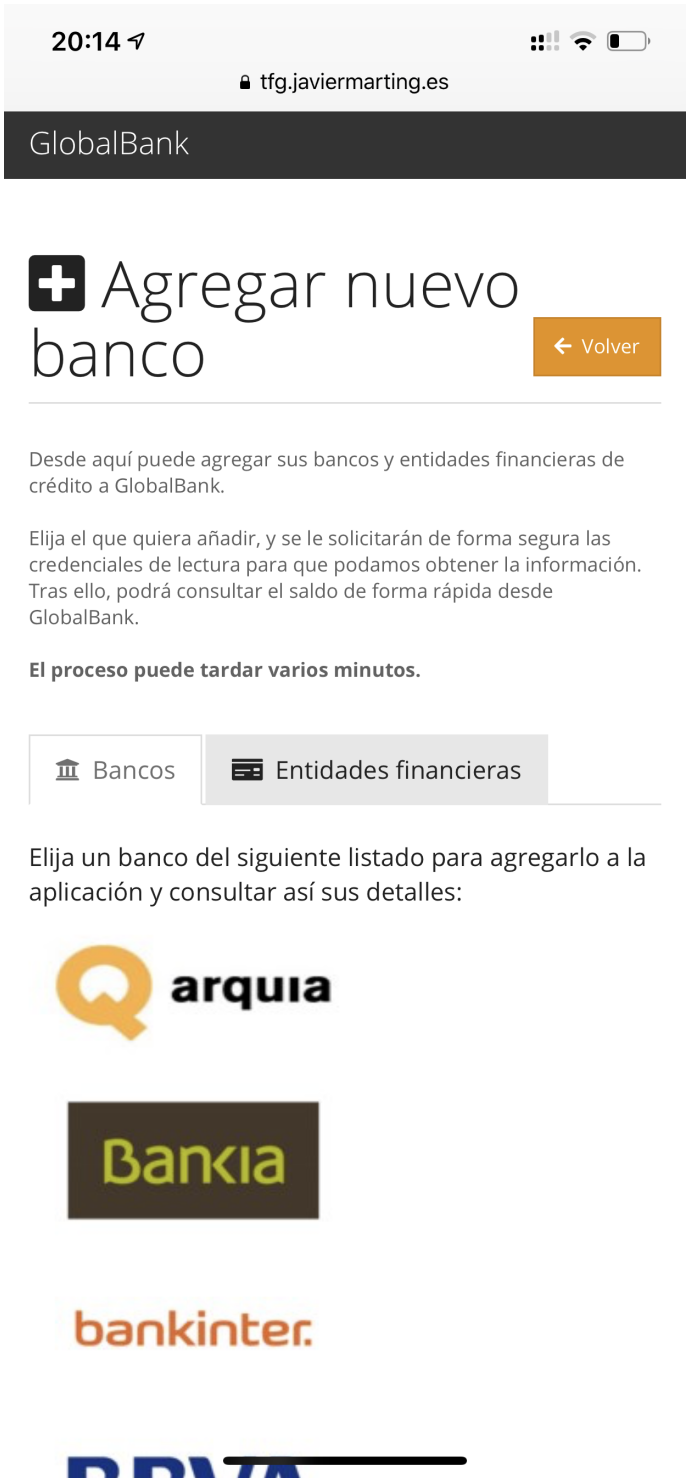

Figura B.9: Visualización de interfaz de agregar banco en pantallas pequeñas